ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

"DISTRIBUCIÓN Y ADMINISTRACIÓN DE CONTENIDOS DE SITIOS WEB UTILIZANDO TÉCNICAS DE REPLICACIÓN Y SINCRONIZACIÓN"

TESIS DE GRADO

Previa a la obtención del Título de:

INGENIERO EN COMPUTACIÓN

ESPECIALIZACIÓN SISTEMAS DE INFORMACIÓN

Presentado por:

PAOLA MIRELLA BONILLA BERMEO

GERARDO GARNICA BENÍTEZ

GUAYAQUIL – ECUADOR

2005

AGRADECIMIENTO

A Dios, motor de nuestras vidas, por todas y cada una de las bendiciones recibidas a lo largo de nuestro caminar.

A nuestros padres por su amor, dedicación y sacrificio continuo en pro de nuestro bienestar y superación.

A nuestros amigos, familiares y demás seres queridos por su cariño y apoyo incondicional.

Al Ing. Fabricio Echeverría, director de nuestra tesis, por compartir sus conocimientos y hacer posible con su colaboración el desarrollo de la misma.

A nuestros apreciados profesores por colaborar en nuestro desarrollo ético y profesional.

DEDICATORIA

A Dios.

A nuestros padres.

A nuestros seres queridos.

TRIBUNAL DE GRADO

ING. MIGUEL YAPUR

Presidente del tribunal

ING. FABRICIO ECHEVERRÍA

Director de tesis

CIB-ESPOL

ING. MARCELO LOOR

Miembro Principal

CIB-ESPOL

ING. KARINA ASTUDILLO

Miembro Principal

CIB-ESPOL

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de esta Tesis de Grado, me corresponde exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL"

(Reglamento de exámenes y títulos profesionales de la ESPOL)

PAOLA BONILLA BERMEO

GERARDO GARNICA BENÍTEZ

RESUMEN

En la actualidad, la forma de trabajar está cambiando veloz y radicalmente y la información es uno de los activos más importantes que poseen las organizaciones, para lo cual utilizan soluciones que cubren dichas necesidades. De esta manera, el reto es presentar estas tecnologías y aplicaciones como un todo unificado y accesible a través del Internet, para que esté disponible en todo momento a diferentes tipos de usuarios.

Igualmente, es importante desarrollar aplicaciones que manejen información dentro de una base de datos distribuida, en donde el contenido no esté centralizado en un solo sitio y pueda ser ingresado y modificado desde el lugar donde se produce la transacción y luego, a través del proceso de sincronización de datos, se actualicen los otros repositorios de datos de la organización.

Este trabajo de tesis presenta una solución que combina el uso de un sistema administrador de contenidos de un negocio de arriendo de inmuebles, que posee varios lugares en donde se almacenan los datos, y que utiliza un sincronizador de datos para distribuir las modificaciones realizadas durante una transacción de un lugar a otro.

En el Capítulo 1 se detalla una introducción básica sobre la importancia de la información y las técnicas para distribuirla; además se explican las justificaciones para el desarrollo de esta tesis y se indica cuáles serían los tipos de negocios que utilizarían este tipo de soluciones.

En el Capítulo 2 se hace un análisis de los requerimientos funcionales y no funcionales, así como de los elementos de la interacción, y las razones por las cuales se seleccionaron las herramientas para la implementación del sistema.

En el Capítulo 3 se detalla el diseño de la arquitectura del sistema, de la base de datos y la interfaz con el usuario. En el Capítulo 4 se explican el proceso de implementación y las pruebas realizadas al sistema.

INDICE GENERAL

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCIÓN

Este proyecto de tesis tiene como objetivo el desarrollo de una solución que administre los contenidos de un sitio Web, y que con el uso de técnicas de sincronización, facilite la transmisión y distribución de la información que se encuentra en diferentes repositorios de datos que poseen la misma estructura de almacenamiento para los mismos. De esta forma se permite a los usuarios acceder a los contenidos y datos del sitio Web desde cualquier lugar donde opere el negocio y paralelamente, éstos, dependiendo de su nivel de acceso, pueden manipular la información del mismo, con la seguridad de que ésta será replicada a todos los puntos de negocio conectados de forma automática.

La solución desarrollada para la demostración de este proyecto de tesis, consiste en un sistema administrador de contenidos de un negocio de alquiler de inmuebles; este negocio mantiene puntos de administración en diferentes lugares, los cuales poseen las mismas estructuras de almacenamiento de datos, y cuyas transacciones se distribuyen a los otros sitios a través de un servicio de sincronización. Este servicio replica la información de cada uno de los registros de las tablas que hayan sido objeto de alguna transacción, luego de la última vez que se ejecutó el servicio. De esta forma se provee una solución que optimiza los recursos del sistema como tiempo y dinero, mediante rapidez y eficacia en la manipulación de la información.

CAPITULO

Antecedentes y justificación

CAPÍTULO 1

ANTECEDENTES Y JUSTIFICACIÓN

1.1. Disponibilidad de la información

Hoy en día realizar negocios desde cualquier lugar del mundo sin el temor de ausentarse de su oficina y no estar en contacto, es una realidad. La necesidad de los administradores de tener siempre disponible la información vital para la toma de decisiones es posible ahora gracias a la tecnología.

De la misma manera, el volumen y distribución geográfica de los datos han ido creciendo en forma geométrica en los últimos años; esto se debe en gran medida a la incorporación de los negocios en el uso de las tecnologías de información, y a cómo el manejo de la información se ha ido distribuyendo en diferentes localidades.

De esta manera, las organizaciones tienen que hacer frente al incremento de las distancias entre los usuarios de la información y al deseo de éstos de obtener un acceso más rápido a los datos.

Es importante tener en cuenta que los datos son improductivos si no pueden administrarse adecuadamente para tenerlos disponibles en el momento y lugar en que se los requieran y que la información es un valor fundamental para las organizaciones que pretendan mantener o incrementar su posición en el mercado en términos de productividad, diferenciación y relevancia.

Así pues, para superar estos obstáculos las organizaciones deben escoger entre:

- Centralizar las aplicaciones y mantener un buen funcionamiento de la red para simplificar el desarrollo y control de las aplicaciones, limitando el alcance de éstas.

- Descentralizar y pasar las aplicaciones a nivel de usuario permitiendo la posibilidad de conectar aplicaciones distribuidas, reduciendo la fiabilidad de la red.

Así, por ejemplo, en pequeños negocios y puntos de ventas es más conveniente utilizar aplicaciones descentralizadas para incrementar su fiabilidad y la satisfacción de los usuarios, pero a la vez exige el uso de un mecanismo de compartición de datos de tal manera que los usuarios de la aplicación puedan obtener datos coherentes y puntuales.

1.2. Técnicas existentes para la distribución de la información

En estos días existen dos técnicas que se pueden emplear para compartir datos dentro de ambientes distribuidos:

- La replicación, y
- La sincronización.

Así pues, en el mercado existen disponibles soluciones de replicación tanto de hardware como de software (Tabla 1.1. Fuente: http://searchstorage.techtarget.com/tip/1,289483,sid5_gci874872,00.html). Varias empresas ofrecen subsistemas de almacenamiento que permiten realizar replicación a través de hardware entre un sitio principal y los sitios de contingencia.

Adicionalmente, es posible encontrar soluciones de replicación a través de software que pueden llegar a abarcar ciertas aplicaciones en forma conjunta, o en su defecto, es posible manejar la replicación nativa de cada motor de bases de datos como, por ejemplo, SQL Server, Oracle o Sybase.

A diferencia de las opciones nombradas anteriormente, el uso de un sincronizador de datos facilita la puesta en producción de la distribución de datos en un ambiente heterogéneo debido a que este esquema trabaja directamente en la capa de datos y no sobre el log[1] como es el caso de la replicación.

[1] El log es un archivo que almacena un registro de las transacciones realizadas en la base de datos.

Soluciones disponibles de replicación de datos	
Basadas en hardware	**Basadas en software**
Sincrónica: • EMC SRDF • IBM PPRC • HP DRM • HDS TrueCopy sync Asincrónica: • HDS TrueCopy async • HP DRM	• Legato Octopus and RepliStor • Topio SANSafe • NSI Doubletake • Veritas VVR • Fujistu Softek TDMF • Peer software's PeerSync • Diasoft File Replication Pro • Insession replication suite • SoftwarePursuits SureSync • XOsoft WANSync • Sunopsis Java Replication Suite

Tabla 1.1. Soluciones disponibles en el mercado de replicación de datos

1.3. Justificación

La principal justificación para el desarrollo de esta tesis es proporcionar una solución que permita obtener información consistente y, sobretodo, justo a tiempo a través de un sitio Web que cuente con un administrador de contenidos; todo esto con el fin de ayudar a la organización de la información de cualquier compañía y optimizar los recursos de las mismas.

Con esta aplicación, se facilitará la transmisión y distribución de contenidos desde un sitio Web a otro cuando éstos sean modificados independientemente de la plataforma en la que se han desarrollado, lo cual asegurará la

portabilidad del contenido. Cabe recalcar que en esta aplicación no se utilizarán componentes distribuidos, con el fin de asegurar la heterogeneidad de las plataformas. Adicionalmente, con esta solución, se disminuye el número de interacciones humanas, y con esto, el número de errores al momento de manipular la información.

Así pues, para la distribución oportuna de los datos se ha escogido la técnica de sincronización, puesto que en la actualidad aporta un enfoque innovador al complejo y tedioso proceso de distribución de la información entre un servidor central y las bases de datos remotas.

1.4. Posibles usuarios del sistema

Este sistema que combina el uso de un administrador de contenidos con sincronización de datos está orientado a los siguientes usuarios:

- Negocios que manejan grandes volúmenes de información, debido a que con el uso de un administrador de contenidos se logra organizar y clasificar la información de la organización y a la vez con la sincronización, se logra una distribución oportuna y con disminución de errores humanos.

- Negocios que poseen sitios Web que necesitan actualización continua y dinámica de la información que presentan, para ello el administrador ofrece una interfaz amigable y fácil de usar para cualquier usuario no técnico que esté capacitado en el manejo del sitio Web.

- Negocios que poseen puntos de venta distribuidos geográficamente, ya que con la sincronización de datos, la distribución de los mismos se haría automáticamente disminuyendo la manipulación humana y con ello, también el número de errores. Cabe recalcar, que se debe tomar en consideración resguardar la consistencia de los datos.

- Negocios que requieran minimizar costos en detalles técnicos de mantenimiento de sus sitios Web, puesto que al emplear un administrador de contenidos, personal no técnico, puede encargarse del manejo del contenido del sitio Web.

CAPITULO

2

Análisis del sistema

CAPÍTULO 2

ANÁLISIS DEL SISTEMA

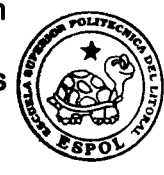
2.1. Análisis de requerimientos y alcance del sistema

En esta sección se realizará el análisis tanto de los requerimientos funcionales y no funcionales referentes al desarrollo del sistema.

2.1.1. Requerimientos funcionales

Aquí se definirán los requerimientos funcionales del sistema, los cuales delimitarán el alcance del proyecto e indicarán el comportamiento observable y la funcionalidad del mismo.

Para efectos de demostración de esta tesis se ha desarrollado un sistema administrador de bienes raíces que, a través de una interfaz Web, permite llevar un control de arriendos de departamentos, villas, oficinas y locales, en el cual se tiene un registro de todos los propietarios y arrendatarios de los inmuebles.



Esta aplicación permite realizar un seguimiento de diferentes procesos del negocio, entre ellos constan los pagos de los arriendos mensuales por parte de los arrendatarios, y una serie de consultas con el fin de simplificar cada uno de dichos procesos.

Además, existe el módulo de administración de contenidos para facilitar las tareas de los usuarios en la manipulación del sitio. Para este efecto se han definido 3 tipos de usuarios:

- Administradores, quienes tienen la capacidad de administrar todo el contenido del sitio, y a su vez manejan los permisos de los usuarios y de los grupos, así como definen las configuraciones de seguridad.

- Agentes operadores de bienes raíces, que son los encargados de la publicación de inmuebles que están disponibles para ser alquilados, así como del ingreso, modificación y eliminación de arrendadores, arrendatarios, garantes e inmuebles. Adicionalmente, son encargados del proceso de generación de contratos y del pago de arriendos.

- Usuarios públicos, quienes tienen acceso a la información del sitio Web, mas no a la manipulación del contenido.

El sistema permite agregar más tipos dependiendo de las necesidades del negocio.

Finalmente, como se desea lograr consolidación de datos en diferentes sitios y justo a tiempo, el sistema proveerá un módulo de sincronización de datos, que será manejado por los administradores del sistema.

2.1.1.1. Especificaciones de funcionalidad

Dentro de la aplicación demostrativa de esta tesis, los agentes operadores de bienes raíces poseen las siguientes funcionalidades:

- **Registros de arrendadores.-** Los agentes pueden ingresar nuevos registros de arrendadores, quienes son los dueños propietarios de uno o más inmuebles, modificar sus datos o eliminarlos del sistema. Los datos que se necesitan para un arrendador son: cédula de identidad, nombres y apellidos, lugar de trabajo y dirección de su domicilio.

- **Registros de arrendatarios.-** Los agentes pueden ingresar nuevos registros de arrendatarios, quienes son las personas que están alquilando uno o más inmuebles, modificar sus datos o eliminarlos del sistema. Los datos que se necesitan para un arrendatario son: cédula de

identidad, nombres y apellidos, lugar de trabajo, nacionalidad y número de cuenta corriente.

- **Registros de garantes.-** Los agentes tienen la capacidad de ingresar nuevos registros de garantes, quienes son las personas que respaldan a los arrendatarios para el alquiler del inmueble, modificar sus datos o eliminarlos del sistema. Los datos que se necesitan para un garante son: cédula de identidad, nombres y apellidos.

- **Registros de inmuebles.-** Los agentes pueden ingresar o editar registros de inmuebles. Los datos que se necesitan para un inmueble son: ciudad, sector y dirección donde se encuentra, tipo de inmueble, propietario, descripción breve, dimensión en metros cuadrados de construcción y terreno, valor del arriendo, número de habitaciones, así como una imagen del inmueble, si es que se tuviera alguna.

- **Contratos.-** Los agentes ingresan la información de los contratos de arriendo de los inmuebles registrados en el sistema. La información que se requiere para un contrato es: datos del arrendador, del arrendatario y del garante, información del inmueble, fecha de suscripción del contrato y monto mensual del arriendo. Cuando se realiza un contrato, automáticamente se generan las letras de cambio mensuales para el pago del arriendo.

- **Registros de ciudades.-** Los agentes pueden ingresar nuevos registros de ciudades al sistema, y modificarlas si se requiere. Se necesita el nombre de la misma para realizar el ingreso.

- **Registros de sectores de ciudades.-** Los agentes pueden ingresar nuevos registros de sectores de una ciudad y modificarlos si se requiere. Los datos que se necesitan para un sector son: ciudad a la que pertenece, y nombre del sector.

- **Registros de tipos de inmueble.-** Los agentes pueden ingresar nuevos registros de tipos de un inmueble y modificarlos si se requiere. Se necesita el nombre del tipo para efectuar el ingreso.

- **Pagos de arriendos.-** Los agentes ingresan el pago mensual de los arriendos de un contrato que realiza el arrendatario de un inmueble.

- **Consulta de inmuebles.-** Los agentes puede consultar cuáles son los inmuebles que están siendo arrendados y los que se encuentran disponibles, pudiendo filtrar la consulta según la ciudad, el sector, el tipo de inmueble, el estado del inmueble (alquilado o no alquilado) y los valores de arriendo.

- **Consulta de pagos de arriendos por arrendatario.**- Los agentes de bienes raíces verifican el estado de los pagos mensuales de los alquileres realizados por un arrendatario específico registrado en el sistema.

- **Consulta de pagos de arriendos por mes.-** Los agentes de bienes raíces pueden realizar consultas por mes de los arriendos que han sido cancelados y los que están adeudados, y que corresponden al mes consultado. Adicionalmente, se debe especificar el año en la consulta.

- **Consulta de personas.-** Los agentes de bienes raíces pueden consultar los listados de arrendadores, arrendatarios y garantes que se encuentran registrados en el sistema.

Por su parte, los administradores del sistema tienen las siguientes atribuciones:

- **Registros de usuarios y grupos de usuarios.-** Los administradores pueden ingresar y modificar usuarios y grupos de usuarios. Es necesario especificar el grupo de usuario al cual va a pertenecer un usuario cuando éste es ingresado en el sistema.

- **Registros de menús.-** Los administradores pueden ingresar y modificar menús del sistema. Es posible que un menú esté contenido dentro de otro, por lo que es necesario especificar a qué menú pertenece si éste fuese el caso.

- **Administración de permisos.-** Los administradores otorgan los permisos de las opciones a los que van a tener acceso cada uno de los grupos de usuarios que se encuentran registrados en el sistema.

- **Administración de formularios.-** Los administradores se encargarán de administrar el contenido de los formularios, ingresando los controles que se necesitan para las operaciones que manejan todos los usuarios.

- **Ingresos.-** Los administradores pueden ingresar ciudades, sectores de ciudades, tipos de inmueble y tipos de listado, al sistema.

- **Administración de elementos del buscador.-** Los administradores pueden ingresar, modificar y eliminar ítems del buscador del sistema. Es necesario especificar el tipo del listado al cual se va a asociar un ítem del buscador.

- **Sincronización.-** Los administradores pueden ingresar y modificar bases de origen y destino y adicionalmente eliminar bases de destino en el sistema. Es necesario agregar una base de origen antes de agregar las bases de destino que se requieran. Por otra parte, es aquí que el usuario configura las tablas del sistema que pueden sincronizar sus datos, agregando o modificando las mismas señalando cuáles son los campos de cada tabla, los campos de comparación, el nivel de prioridad de cada tabla, y otros parámetros de configuración.

Finalmente, los usuarios públicos del sistema pueden hacer:

- **Ver ofertas recientes.-** Permite observar una vista general de los inmuebles que están disponibles más recientemente en el sistema, y consultarlos individualmente cada uno de estos.

- **Consulta de inmuebles disponibles.-** Los navegadores pueden consultar cuáles son los inmuebles disponibles según el tipo y la ubicación donde se encuentran. Para este tipo de consulta se puede utilizar el buscador o el menú Alquiler, especificando el tipo de inmueble que se requiera consultar.

2.1.2. Requerimientos no funcionales

En esta sección se definen los requerimientos que debe cumplir el sistema en cuanto a confiabilidad y seguridad.

- Las transacciones realizadas en el sistema, como ingresos y pagos de arriendos deben estar correctamente validadas para que no se produzcan más de una vez.

- Los datos presentados a los usuarios deben ser correctos y no deben prestarse a errores que puedan confundir a los usuarios.

- El servicio de sincronización debe realizarse periódicamente, según como se encuentre configurado dentro de los servidores que sirven como publicadores del sistema. También es recomendable poseer respaldos periódicos de las bases de datos que están sincronizadas entre sí.

- El servicio de sincronización debe llevar un registro de errores cuando falle una inserción o actualización de datos en una tabla de la base que está siendo sincronizada.

2.2. Análisis de tecnologías

El desarrollo de este proyecto involucra el desarrollo de dos aspectos: el manejo de información dinámica del sitio Web, y el desarrollo del servicio de sincronización de datos.

En el campo de las tecnologías Web, éstas se pueden dividir en dos grupos: tecnologías del lado cliente y tecnologías del lado servidor. En ocasiones la misma tecnología puede usarse en uno u otro lado con ligeras variaciones, o mezclarlas entre sí. Las tecnologías del lado cliente están orientadas a ejecutarse en los nodos cliente. Sin embargo, para el uso de páginas con contenido que cambia dinámicamente, es necesario el uso de las tecnologías del lado servidor.

Los lenguajes de servidor permiten que las páginas dinámicas se escriban en el mismo archivo mezclado con código HTML[2]. Cuando una página es solicitada por parte de un cliente, el servidor ejecuta los scripts y se genera una página resultado, que solamente contiene código HTML, tal como se muestra en la figura 2.1. Este resultado final es el que se envía al cliente y puede ser interpretado sin lugar a errores ni incompatibilidades, puesto que sólo contiene HTML. El servidor Web es el que administra la información de las bases de datos y cualquier otro recurso, como imágenes o servidores de

[2] Hypertext Markup Language, es un lenguaje utilizado para crear documentos, en especial para la Web, compatibles con varias plataformas.

correo, y luego envía al cliente una página Web con los resultados de todas las operaciones ejecutadas con algún lenguaje del lado servidor [11].

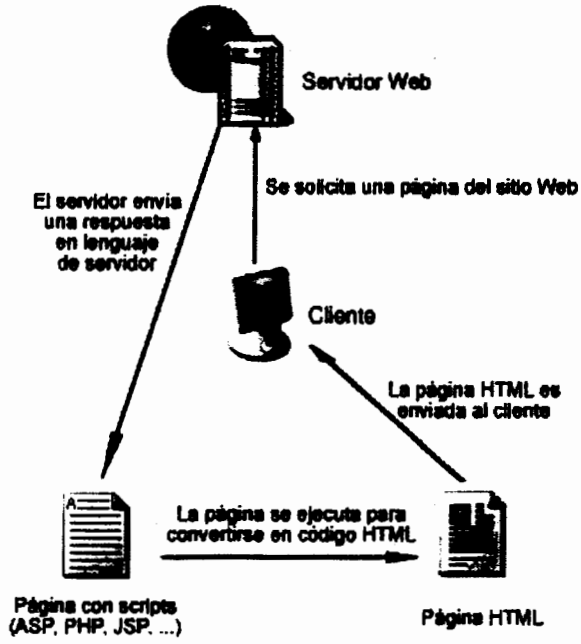


Figura 2.1. Funcionamiento de tecnologías Web del lado servidor

La desventaja de su uso es que se necesita un servidor más potente y con más capacidades que el necesario para simples páginas codificadas en HTML. Además, posiblemente, estos servidores podrán soportar menos usuarios concurrentes, porque se requerirá más tiempo de procesamiento para cada uno.

A continuación se detallan las características, ventajas y desventajas de las principales tecnologías servidor:

CGI (Common Gateway Interface).- Es el sistema más antiguo que existe para la programación de las páginas dinámicas de servidor. Actualmente se encuentra un poco desfasado por diversas razones, entre las que destaca la dificultad con la que se desarrollan los programas y la pesada carga que supone para el servidor que los ejecuta. Los CGI se escriben habitualmente en el lenguaje Perl, sin embargo, otros lenguajes como C, C++, o Visual Basic pueden también ser empleados. Entre las ventajas de usar CGI podemos destacar que se pueden desarrollar con cualquier lenguaje, son muy estables y están altamente extendidos. Entre las desventajas de su uso tenemos que cada acceso implica la creación de un nuevo proceso en el servidor, degradando su rendimiento, debido a que se tendrán distintos accesos a un mismo recurso creando varios procesos simultáneos.

ASP (Active Server Pages).- Es la tecnología desarrollada por Microsoft que se escribe en la misma página Web, utilizando el lenguaje VBScript o JScript, y con el cual se pueden realizar muchos tipos de aplicaciones distintas como acceso a bases de datos, al sistema de archivos del servidor y en general a todos los recursos que tenga el propio servidor. Entre sus ventajas cuenta con que ofrece la ventaja de utilizar el concepto de objetos COM (Component Object Model), que son objetos en algún otro lenguaje (ej.: ejecutables en C++ o Java), de manera que si ya se tiene algún programa las páginas ASP pueden hacer uso de los métodos en estos objetos.

La principal desventaja con la que choca esta tecnología es que sólo se puede usar en plataformas Microsoft, con el consiguiente problema de portabilidad que esto presenta. El tipo de servidor Web que emplea ASP es Internet Information Server o Personal Web Server. Para conectarse a una base de datos, normalmente se utiliza ADO, que es un adaptador universal a bases de datos que se especializa posteriormente para comunicarse con una base de datos concreta.

PHP (Hypertext Preprocessor).- Sistema OpenSource[3] desarrollado originalmente en el año 1994 por Rasmus Lerdorf como un CGI escrito en C que permitía la interpretación de un número limitado de comandos. Es de libre distribución y desde su versión 4, pese a ser un lenguaje interpretado, es bastante rápido. Es sencillo, de sintaxis cómoda y fácil, se ejecuta en un único espacio de memoria y esto evita las comunicaciones entre componentes COM que se realizan en otras tecnologías como ASP. Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones Web y tiene compatibilidad con las bases de datos más comunes.

Es independiente de plataforma, puesto que existe un módulo de PHP para casi cualquier servidor Web, como Apache o IIS, lo que permite que cualquier sistema pueda ser compatible con el lenguaje; lo que también significa una ventaja importante, ya que permite portar el sitio desarrollado en PHP de un sistema a otro sin prácticamente ningún trabajo. Como desventaja del uso

[3] Los programas de código abierto (OpenSource) permiten que el código fuente esté disponible para que los programadores lo puedan obtener y modificar.

de PHP podemos citar que se han producido algunos errores de seguridad (bugs), aunque estos han sido solucionados por su comunidad de desarrolladores.

Perl.- Es rápido para hacer una pequeña aplicación Web. Posee cientos de bibliotecas operativas y listas para ser usadas. Perl debe ser usado precompilado vía ModPerl o el impacto en procesador es muy alto.

Servlets.- Es una aplicación del tipo CGI realizada en Java y que tiene métodos para atender requerimientos. El servidor pasa las variables hacia y desde el ambiente de ejecución del Servlet. Hace uso de hilos en vez de procesos para distintas solicitudes a un mismo servicio y uso de sesiones.

JSP (Java Server Pages).- Utiliza código Java junto con HTML, es parecido al resto de lenguajes en sintaxis (PHP y ASP), presenta separación entre presentación y contenido, permite reusar componentes (JavaBeans) en distintas plataformas (UNIX, Windows), así como usa XML[4] en los scripts. El motor de las páginas JSP está basado en los servlets de Java, y en JSP se generan archivos que incl22uyen, dentro de la estructura de etiquetas HTML, las sentencias Java que se van a ejecutar en el servidor. Antes de que sean funcionales los archivos, el motor JSP lleva a cabo una fase de traducción de esa página en un servlet.

[4] XML (eXtensible Markup Language), es un lenguaje que proporciona un formato para describir datos estructurados, lo que permite desarrollar nuevas aplicaciones de presentación y manipulación de datos basadas en Web.

Las aplicaciones que usan JSP tienen un mantenimiento más fácil que otras, ya que Java es un lenguaje estructurado y es más fácil de construir. Además, la tecnología JSP hace mayor énfasis en los componentes que en los scripts, lo que hace que sea más fácil revisar el contenido sin que se afecte a la lógica o revisar la lógica sin cambiar el contenido. Debido a que JSP es abierto y multiplataforma, l22os servidores Web, plataformas y otros componentes pueden ser fácilmente actualizados o cambiados sin que afecte a las aplicaciones basadas en la tecnología JSP.

Características	ASP	PHP	JSP	CGI
Plataforma	Windows	Windows, Linux, Solaris,	Windows, Linux, Solaris, etc.	Windows, Linux, Solaris, etc.
Servidor Web	Internet Information Server, Personal Web Server (ChiliSoft ASP permite ejecutar ASP en otros ambientes)	Apache, IIS, Netscape, Xitami, etc.	Apache, IIS, Netscape, etc.	Apache, IIS, Netscape, etc.
Lenguaje de programación	VBScript, JScript, Perl	Similar a C	Java	Perl, C, Java
Integración de orígenes de datos	ODCB, ADO, OLEDB	ODBC	ODBC, JDBC	ODBC
Facilidad de aprendizaje	Rápida	Rápida	Media, con conocimientos en Java	Media
Portabilidad	No	Sí	Sí	Sí
Desempeño	Bueno	Bueno, mejor que ASP	Bueno, mejor que ASP	Regular

Tabla 2.1. Tabla comparativa de las tecnologías Web de lado servidor

La tabla 2.1 presenta un análisis comparativo de las principales tecnologías del lado servidor, resumiendo lo expuesto con anterioridad.

Para realizar este proyecto de tesis se ha optado por el uso de PHP debido a su flexibilidad, facilidad de uso y mantenimiento, e independencia de plataforma, lo cual cumplirá con uno de los objetivos del sistema de desarrollar una aplicación que sincronice datos independientemente de la plataforma que se esté utilizando. Además permite la implementación de clases como cualquier lenguaje de programación orientado a objetos, lo que permite encapsular la lógica de negocios de una aplicación y separarla del contenido.

Por otro lado, para el desarrollo del servicio de transferencia de datos entre diferentes nodos se ha considerado crear un mecanismo de sincronización para la transferencia de datos entre las diferentes bases de datos del sistema distribuido debido a que este esquema, a diferencia de la replicación de datos que trabaja con el log de cada motor de base de datos, funciona directamente en la capa de datos, por lo que su uso reduce la complejidad en el desarrollo de la aplicación aunque aumentan las responsabilidades en el manejo de la seguridad de los datos.

Para tal fin se ha optado por desarrollar un mecanismo de sincronización de datos distribuido que se comunique a través de una red (que puede ser LAN[5] o WAN[6]), en donde cada uno de los puntos repositorios de datos del modelo

[5] Red de área local (Local Area Network).
[6] Red de área extensa (Wide Area Network).

distribuido mantiene su información y envían a los otros repositorios las transacciones que se van produciendo tal como se muestra en la Figura 2.2.

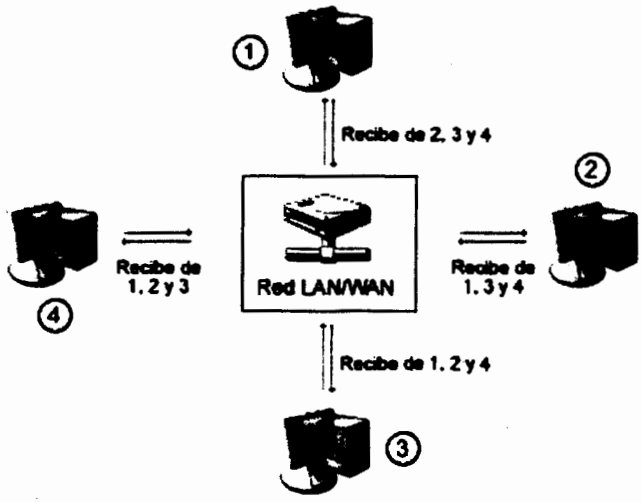


Figura 2.2. Esquema de sincronización distribuida:
Los nodos envían y reciben datos entre ellos directamente

Las reglas que permiten el buen funcionamiento de la sincronización de datos deben quedar almacenadas dentro de los perfiles de cada punto:

- Si puede o no modificar la información de otros puntos.
- Cuáles son los puntos de donde se recibe información.
- Cuál es el tipo de información se envía.
- Si de un punto concreto se debe recibir datos específicos.
- En qué lugar se depositan sus variaciones: espacio Web, computadora portátil, etc.

Entre las ventajas del uso de un sincronizador de datos podemos mencionar:

- Produce un mejor desempeño de las aplicaciones ya que éstas pueden operar sobre copias de datos locales en vez de tener que comunicarse con sitios remotos.

- Significa una mejor disponibilidad de los datos ya que de todo objeto está disponible por lo menos una copia, esencial sobretodo en el caso de que se necesite recuperar información.

- Permite partir la base de datos existente en múltiple servidores cuando se fragmentan los datos que se encuentran en un solo servidor. Con esta capacidad se puede optimizar el uso de la red, procesando y almacenando los datos en diferentes servidores.

2.3. Descripción de casos de uso y escenarios

Debido a que uno de los propósitos de este proyecto de tesis es desarrollar una aplicación que permita administrar contenidos y sincronizar los datos entre diferentes repositorios, se ha optado por realizar un sistema orientado a objetos en el cual los objetos encapsulen el comportamiento del sistema a través de sus atributos y operaciones.

Para conseguir este objetivo se va a utilizar UML (*Unified Modeling Language* – Lenguaje de Modelamiento Unificado) el cual es un lenguaje de modelamiento de propósito general diseñado para especificar, visualizar y documentar los elementos de un sistema [5].

La notación UML proporciona casos de uso y escenarios para el modelamiento de un sistema. Un caso de uso es el conjunto de transacciones en un sistema cuyo objetivo es producir un resultado con valor medible.

Los escenarios describen los eventos a los que el sistema responderá para proveer la funcionalidad descrita en el caso de uso.

A continuación se describen los casos de uso para el sistema administrador de bienes raíces:

1. Usuario ingresa al sistema.
2. Agente ingresa arrendador.
3. Agente ingresa arrendatario.
4. Agente ingresa garante.
5. Agente ingresa inmueble.
6. Agente ingresa contrato.
7. Agente o administrador ingresa ciudad.
8. Agente o administrador ingresa sector de ciudad.
9. Agente o administrador ingresa tipo de inmueble.

10. Agente registra pago de arriendo.
11. Agente realiza consulta de inmuebles.
12. Agente realiza consulta de pagos.
13. Agente realiza consulta de personas.
14. Agente modifica datos de arrendador.
15. Agente modifica datos de arrendatario.
16. Agente modifica datos de garante.
17. Agente modifica datos de inmueble.
18. Agente o administrador modifica ciudad.
19. Agente o administrador modifica sector de ciudad.
20. Agente o administrador modifica tipo de inmueble.
21. Agente elimina arrendador.
22. Agente elimina arrendatario.
23. Agente elimina garante.
24. Agente elimina contrato.
25. Administrador ingresa menú.
26. Administrador ingresa grupo de usuario.
27. Administrador ingresa usuario.
28. Administrador ingresa control de formulario.
29. Administrador otorga permisos.
30. Administrador ingresa tipo de listado.
31. Administrador modifica datos de menú.
32. Administrador modifica datos de grupo de usuario.

33. Administrador modifica datos de usuario.
34. Administrador ingresa base de origen del sincronizador.
35. Administrador ingresa base de destino del sincronizador.
36. Administrador ingresa tabla del sistema.
37. Administrador modifica tabla del sistema.
38. Administrador agrega ítem del buscador.
39. Administrador modifica ítem del buscador.
40. Administrador elimina ítem del buscador.
41. Usuario público busca inmuebles.

Nombre: 1. Usuario ingresa al sistema.

Descripción: Un agente o administrador del sistema intenta ingresar al sistema a través de su nombre de usuario y contraseña.

Escenarios:

1.1. Acceso autorizado a un administrador del sistema.

1.2. Acceso autorizado a un agente de bienes raíces.

1.3. Acceso no autorizado a una persona cuyo nombre de usuario es incorrecto.

1.4. Acceso no autorizado a una persona cuyo nombre de usuario es correcto, pero la contraseña es inválida.

1.5. Acceso no autorizado por error de procesamiento en la base de datos del sistema.

Nombre: 2. Agente ingresa arrendador.

Descripción: Un agente ingresa los datos de un nuevo registro de arrendador o propietario de un inmueble.

Escenarios:

2.1. El ingreso de un nuevo arrendador se produce satisfactoriamente.

2.2. No se produce el ingreso porque ya existe el número de cédula del arrendador.

2.3. No se produce el ingreso porque el número de cédula no posee 10 dígitos.

2.4. No se produce el ingreso porque no se han ingresado todos los datos obligatorios.

Nombre: 3. Agente ingresa arrendatario.

Descripción: Un agente ingresa los datos de un nuevo registro de arrendatario o inquilino de un inmueble.

Escenarios:

3.1. El ingreso de un nuevo arrendatario se produce satisfactoriamente.

3.2. No se produce el ingreso porque ya existe el número de cédula del arrendatario.

3.3. No se produce el ingreso porque el número de cédula no posee 10 dígitos.

3.4. No se produce el ingreso porque no se han ingresado todos los datos obligatorios.

Nombre: 4. Agente ingresa garante.

Descripción: Un agente ingresa los datos de un nuevo registro de un garante de contrato.

Escenarios:

4.1. El ingreso de un nuevo garante se produce satisfactoriamente.

4.2. No se produce el ingreso porque ya existe el número de cédula del garante.

4.3. No se produce el ingreso porque el número de cédula no posee 10 dígitos.

4.4. No se produce el ingreso porque no se han ingresado todos los datos obligatorios.

Nombre: 5. Agente ingresa inmueble.

Descripción: Un agente ingresa los datos de un nuevo inmueble dentro del sistema.

Notas:

- El propietario del inmueble debe estar registrado en el sistema.
- El valor del arriendo debe estar en dólares (US$).

Escenarios:

5.1. El ingreso de un nuevo inmueble se produce satisfactoriamente.

5.2. No se produce el ingreso porque no se ha seleccionado la ciudad en la que se encuentra.

5.3. No se produce el ingreso porque no se han ingresado todos los datos obligatorios.

5.4. No se produce el ingreso porque no se han ingresado las dimensiones del inmueble.

5.5. Se produce el ingreso del inmueble, pero no de la imagen asociada debido a que el archivo no tiene la extensión correcta.

5.6. Se produce el ingreso del inmueble, pero no de la imagen asociada debido a que el archivo supera el tamaño máximo permitido.

Nombre: 6. Agente ingresa contrato.

Descripción: Un agente ingresa los datos de un nuevo contrato de arrendamiento de un inmueble.

Notas:

- El ingreso de un nuevo contrato debe generar las letras de cambio mensuales para el pago del arriendo.
- El valor del arriendo debe estar en dólares (US$).

Escenarios:

6.1. El ingreso de un nuevo contrato se produce satisfactoriamente.

6.2. No se produce el ingreso porque no es válida la fecha de suscripción.

6.3. No se produce el ingreso porque el valor del monto del arriendo no es válido.

Nombre: 7. Agente o administrador ingresa ciudad.

Descripción: Un agente ingresa los datos de un nuevo registro de ciudad.

Escenarios:

7.1. El ingreso de una nueva ciudad se produce satisfactoriamente.

7.2. No se produce el ingreso porque ya existe el nombre de la ciudad.

7.3. No se produce el ingreso porque no se ingresó el nombre de la ciudad.

Nombre: 8. Agente o administrador ingresa sector de ciudad.

Descripción: Un agente ingresa los datos de un nuevo registro de un sector de ciudad.

Notas:

- Un sector debe estar asociado a una ciudad.

Escenarios:

8.1. El ingreso de un nuevo sector de ciudad se produce satisfactoriamente.

8.2. No se produce el ingreso porque ya existe el nombre del sector en la ciudad indicada.

8.3. No se produce el ingreso porque no se ha indicado la ciudad.

8.4. No se produce el ingreso porque no se ingresó el nombre del sector.

Nombre: 9. Agente o administrador ingresa tipo de inmueble.

Descripción: Un agente ingresa los datos de un nuevo registro de tipo de inmueble.

Escenarios:

9.1. El ingreso de un nuevo tipo de inmueble se produce satisfactoriamente.

9.2. No se produce el ingreso porque ya existe el nombre del tipo de inmueble.

9.3. No se produce el ingreso porque no se ha indicado el nombre del tipo de inmueble.

Nombre: 10. Agente registra pago de arriendo.

Descripción: Un agente ingresa el pago del alquiler mensual de un inmueble.

Notas:

- Se muestran los valores cancelados y por cancelar que corresponden a un contrato.
- Se indica la fecha de vencimiento de cada una de las letras de cambio.
- Se especifica cuál es la letra de cambio que se va a pagar.

Escenarios:

10.1. El pago del arriendo mensual se produce satisfactoriamente.

10.2. No se presenta ninguna información porque ya no existen letras pendientes de pago.

Nombre: 11. Agente realiza consulta de inmuebles.

Descripción: Un agente consulta los inmuebles disponibles o arrendados dentro de una ciudad y sector, según el tipo de inmuebles o según los valores de arriendo.

Notas:

- Se muestra el nombre del propietario.
- Se muestra la dirección del inmueble.
- Se muestra la descripción del inmueble.
- Se muestra el valor del arriendo de inmueble.

Escenarios:

11.1. La consulta de inmuebles se produce satisfactoriamente.

11.2. No se presentan resultados porque no se ha ingresado ningún criterio de consulta.

Nombre: 12. Agente realiza consulta de pagos.

Descripción: Un agente consulta los pagos que realizados y que están siendo adeudados por parte de un arrendatario, o en un mes específico.

Notas:

- Se muestra el nombre del arrendatario.
- Se muestra la fecha de vencimiento del pago.
- Se muestra el estado de la letra de cambio mensual.
- Se muestra el valor del arriendo mensual.

Escenarios:

12.1. La consulta de pagos se produce satisfactoriamente.

12.2. No se presentan resultados porque el arrendatario no tiene registros de pagos en el sistema.

12.3. No se presentan resultados porque en el mes consultado no existen datos para presentar.

Nombre: 13. Agente realiza consulta de personas.

Descripción: Un agente consulta los arrendadores, arrendatarios y solicitantes registrados en el sistema.

Notas:

- Se muestra el número de identificación de la persona.
- Se muestra el nombre de la persona.
- Se muestra el lugar de trabajo de la persona.
- Se muestra la dirección de domicilio del arrendador.
- Se muestra la nacionalidad y el número de cuenta corriente del arrendatario.

Escenarios:

13.1. La consulta de personas se produce satisfactoriamente.

13.2. No se presentan resultados porque no existen datos de arrendadores para presentar.

13.3. No se presentan resultados porque no existen datos de arrendatarios para presentar.

13.4. No se presentan resultados porque no existen datos de solicitantes para presentar.

Nombre: 14. Agente modifica datos de arrendador.

Descripción: Un agente modifica los datos de un registro de arrendador de un inmueble.

Notas:

- No se puede cambiar el número de cédula del arrendador.
- Se muestran todos los datos del arrendador.

Escenarios:

14.1. La modificación de datos del arrendador se produce satisfactoriamente.

14.2. No se presentan datos porque no existe el número de cédula del arrendador previamente ingresado.

14.3. No se produce la modificación de datos del arrendador porque no se han ingresado todos los datos obligatorios.

Nombre: 15. Agente modifica datos de arrendatario.

Descripción: Un agente modifica los datos de un registro de arrendatario de un inmueble.

Notas:

- No se puede cambiar el número de cédula del arrendatario.
- Se muestran todos los datos del arrendatario.

Escenarios:

15.1. La modificación de datos de un arrendatario se produce satisfactoriamente.

15.2. No se presentan datos porque no existe el número de cédula del arrendatario previamente ingresado.

15.3. No se produce la modificación de datos del arrendatario porque no se han ingresado todos los datos obligatorios.

Nombre: 16. Agente modifica datos de garante.

Descripción: Un agente modifica los datos de un registro de garante.

Notas:

- No se puede cambiar el número de cédula del garante.
- Se muestran todos los datos del garante.

Escenarios: .

16.1. La modificación de datos del garante se produce satisfactoriamente.

16.2. No se presentan datos porque no existe el número de cédula del garante previamente ingresado.

16.3. No se produce la modificación de datos del garante porque no se han ingresado todos los datos obligatorios.

Nombre: 17. Agente modifica datos de inmueble.

Descripción: Un agente modifica los datos de un registro de inmueble.

Notas:

- Se muestran todos los datos del inmueble.

Escenarios:

17.1. La modificación de datos del inmueble se produce satisfactoriamente.

17.2. No se presentan datos porque no existe el ID del inmueble previamente ingresado.

17.3. No se produce la modificación de datos del inmueble porque no se han ingresado todos los datos obligatorios.

Nombre: 18. Agente o administrador modifica ciudad.

Descripción: Un agente o un administrador modifican los datos de un registro de ciudad.

Escenarios:

18.1. La modificación de datos de la ciudad se produce satisfactoriamente.

18.2. No se produce la modificación de datos del garante porque no se ha ingresado el nuevo nombre de la ciudad.

Nombre: 19. Agente o administrador modifica sector de ciudad.

Descripción: Un agente o un administrador modifican los datos de un registro de sector de ciudad.

Escenarios:

19.1. La modificación de datos del sector de ciudad se produce satisfactoriamente.

19.2. No se produce la modificación de datos del sector porque no se ha ingresado el nuevo nombre.

Nombre: 20. Agente o administrador modifica tipo de inmueble.

Descripción: Un agente o un administrador modifican los datos de un registro de tipo de inmueble.

Escenarios:

20.1. La modificación de datos de un tipo de inmueble se produce satisfactoriamente.

20.2. No se produce la modificación de datos del tipo de inmueble porque no se ha ingresado el nuevo nombre.

Nombre: 21. Agente elimina arrendador.

Descripción: Un agente elimina un registro de arrendador.

Notas:

- Se muestran todos los datos del arrendador.

Escenarios:

21.1. La eliminación del registro de arrendador se produce satisfactoriamente.

21.2. No se presentan datos porque no existe el número de cédula del arrendador previamente ingresado.

21.3. No se produce la eliminación por algún error en la base de datos.

Nombre: 22. Agente elimina arrendatario.

Descripción: Un agente elimina un registro de arrendatario.

Notas:

- Se muestran todos los datos del arrendatario.

Escenarios:

22.1. La eliminación del registro de arrendatario se produce satisfactoriamente.

22.2. No se presentan datos porque no existe el número de cédula del arrendatario previamente ingresado.

22.3. No se produce la eliminación por algún error en la base de datos.

Nombre: 23. Agente elimina garante.

Descripción: Un agente elimina un registro de garante.

Notas:

- Se muestran todos los datos del garante.

Escenarios:

23.1. La eliminación del registro de garante se produce satisfactoriamente.

23.2. No se presentan datos porque no existe el número de cédula del garante previamente ingresado.

23.3. No se produce la eliminación por algún error en la base de datos.

Nombre: 24. Agente elimina contrato.

Descripción: Un agente elimina un registro de contrato.

Notas:

- Se muestran todos los datos del contrato.

- Al eliminarse un contrato, todos los pagos pendientes relacionados con este contrato son anulados.

Escenarios:

24.1. La eliminación del contrato se produce satisfactoriamente.

24.2. No se produce la eliminación porque el contrato ya ha finalizado.

24.3. No se presentan datos porque no existe el número de contrato previamente ingresado.

24.4. No se produce la eliminación por algún error en la base de datos.

Nombre: 25. Administrador ingresa menú.

Descripción: Un administrador ingresa los datos de un nuevo registro de menú del sistema.

Notas:

- El menú es el grupo de opciones que tienen los usuarios del sistema.
- Un menú puede estar contenido dentro de otra opción de menú.
- El título del menú es la cabecera que va a aparecer en el área de trabajo.

Escenarios:

25.1. El ingreso de un nuevo menú se produce satisfactoriamente.

25.2. No se produce el ingreso porque ya existe el nombre del menú ingresado dentro de la opción que lo contiene.

25.3. No se produce el ingreso porque no se han ingresado todos los datos obligatorios.

Nombre: 26. Administrador ingresa grupo de usuario.

Descripción: Un administrador ingresa los datos de un nuevo registro de grupo de usuarios del sistema.

Notas:

- El grupo de usuarios indica las opciones a las cuales van a tener acceso los usuarios del sistema.

Escenarios:

26.1. El ingreso de un nuevo grupo de usuarios se produce satisfactoriamente.

26.2. No se produce el ingreso porque ya existe el nombre del grupo de usuarios ingresado.

26.3. No se produce el ingreso porque no se ha ingresado una descripción del grupo de usuarios.

Nombre: 27. Administrador ingresa usuario.

Descripción: Un administrador ingresa los datos de un nuevo registro de usuario del sistema.

Notas:

- El usuario debe pertenecer a un grupo de usuarios del sistema.

Escenarios:

27.1. El ingreso de un nuevo usuario se produce satisfactoriamente.

27.2. No se produce el ingreso porque no se ha especificado el grupo de usuarios.

27.3. No se produce el ingreso porque ya existe el nombre de usuario especificado.

27.4. No se produce el ingreso porque no se ha especificado la contraseña del usuario.

27.5. No se produce el ingreso porque la confirmación de la contraseña no coincide con la contraseña ingresada.

Nombre: 28. Administrador ingresa control de formulario.

Descripción: Un administrador ingresa los datos un control de un formulario del sistema, indicando el menú y submenú en el cual se va a ingresar, así como el tipo de control que va a ser ingresado.

Notas:

- El tipo de control especifica los controles Web soportados por el sistema.

Escenarios:

28.1. El ingreso del control de un formulario se produce satisfactoriamente.

28.2. No se produce el ingreso porque no se ha especificado el grupo de menú.

28.3. No se produce el ingreso porque ya existe el control dentro del formulario especificado.

28.4. No se produce el ingreso porque no se ingresan todos los campos obligatorios.

Nombre: 29. Administrador otorga permisos.

Descripción: Un administrador otorga o deniega permisos a los usuarios, indicando el grupo de usuarios y el grupo de menús en el cual van a ser dados los permisos.

Notas:

- Los permisos otorgados son aquellas operaciones que pueden ejecutar los usuarios dependiendo del grupo de usuarios al que pertenezcan.

Escenarios:

29.1. La operación de otorgar y denegar permisos se ha realizado satisfactoriamente.

29.2. No se presentan los permisos porque no se ha seleccionado el grupo de menús.

Nombre: 30. Administrador ingresa tipo de listado.

Descripción: Un agente ingresa los datos de un nuevo registro de listado.

Notas:

- El tipo de listado es el menú de tipo combo (ComboBox) que va a tener el sistema.

Escenarios:

30.1. El ingreso de un nuevo listado se produce satisfactoriamente.

30.2. No se produce el ingreso porque ya existe el nombre del control.

30.3. No se produce el ingreso porque no se han ingresado los campos obligatorios.

Nombre: 31. Administrador modifica datos de menú.

Descripción: Un administrador modifica los datos de un registro de menú previamente seleccionado.

Notas:

- Se muestran los datos del menú.

Escenarios:

31.1. La modificación de datos del menú se produce satisfactoriamente.

31.2. No se produce la modificación de datos del menú porque no se han ingresado todos los datos obligatorios.

31.3. No se produce la modificación de datos del menú porque ya existe el nombre del menú ingresado.

Nombre: 32. Administrador modifica datos de grupo de usuarios.

Descripción: Un administrador modifica los datos de un registro de grupo de usuarios previamente seleccionado.

Notas:

- Se muestran los datos del grupo de usuarios.

Escenarios:

32.1. La modificación de datos del grupo de usuarios se produce satisfactoriamente.

32.2. No se produce la modificación de datos del grupo de usuarios porque ya existe el nombre del grupo ingresado.

32.3. No se produce la modificación de datos del grupo de usuarios porque no se han ingresado todos los datos obligatorios.

Nombre: 33. Administrador modifica datos de usuario.

Descripción: Un administrador modifica los datos de un registro de usuario previamente seleccionado.

Notas:

- Se muestran los datos del usuario.
- Se debe indicar si se va a ingresar una nueva contraseña.

Escenarios:

33.1. La modificación de datos del usuario se produce satisfactoriamente.

33.2. No se produce la modificación de datos del usuario porque no se ha especificado el grupo de usuarios.

33.3. No se produce la modificación de datos del usuario porque la contraseña ingresada no es la correcta.

33.4. No se produce la modificación de datos del usuario porque la nueva contraseña es la misma que la anterior.

Nombre: 34. Administrador ingresa base de origen del sincronizador.

Descripción: Un administrador ingresa los datos para la configuración de la base de origen desde la cual se van a sincronizar los datos, especificando el host, el puerto, el nombre de la base de datos, y el nombre de usuario y contraseña para acceder a la misma.

Notas:

- El puerto por defecto es el 5432.

Escenarios:

34.1. El ingreso de la base de origen se produce satisfactoriamente.

34.2. No se produce el ingreso porque ya existe una base de datos de origen dentro del archivo de configuración del servicio de sincronización.

34.3. No se produce el ingreso de la base de datos de origen porque no se ingresan todos los campos obligatorios.

Nombre: 35. Administrador ingresa base de destino del sincronizador.

Descripción: Un administrador ingresa los datos para la configuración de una de las bases de destino de los datos que están siendo sincronizados, especificando el host, el puerto, el nombre de la base de datos, y el nombre de usuario y contraseña para acceder a la misma.

Notas:

- Es necesario tener especificada una base de origen antes de ingresar una base de destino.
- El puerto por defecto es el 5432.

Escenarios:

35.1. El ingreso de la base de destino se produce satisfactoriamente.

35.2. No se produce el ingreso porque no existe una base de datos de origen en la configuración del servicio de sincronización.

35.3. No se produce el ingreso porque ya existe una base de datos de destino con las especificaciones indicadas.

35.4. No se produce el ingreso de la base de datos de destino porque no se ingresan todos los campos obligatorios.

Nombre: 36. Administrador ingresa tabla del sistema.

Descripción: Un administrador ingresa los datos para la configuración de una tabla que tiene la capacidad de sincronizar sus datos hacia otros puntos del sistema.

Escenarios:

36.1. El ingreso de la tabla se produce satisfactoriamente.

36.2. No se produce el ingreso porque ya existe el nombre de la tabla.

36.3. No se produce el ingreso de la tabla porque no se ingresan todos los campos obligatorios.

Nombre: 37. Administrador modifica tabla del sistema.

Descripción: Un administrador modifica los datos de un registro de tabla previamente seleccionado.

Notas:

- Se muestran los datos de la tabla del sistema.

Escenarios:

37.1. La modificación de datos de la tabla del sistema se produce satisfactoriamente.

37.2. No se produce la modificación de datos de la tabla porque no se han ingresado todos los datos obligatorios.

Nombre: 38. Administrador agrega ítem del buscador.

Descripción: Un administrador ingresa los datos de un nuevo ítem del buscador del sistema.

Escenarios:

38.1. El ingreso del ítem del buscador se produce satisfactoriamente.

38.2. No se produce el ingreso porque ya existe la descripción del ítem.

38.3. No se produce el ingreso del ítem porque no se ingresan todos los campos obligatorios.

Nombre: 39. Administrador modifica ítem del buscador.

Descripción: Un administrador modifica los datos de un registro de ítem del buscador previamente seleccionado.

Notas:

- Se muestran los datos de un ítem del buscador.

Escenarios:

39.1. La modificación de datos de un ítem del buscador se produce satisfactoriamente.

39.2. No se produce la modificación de datos del ítem porque no se han ingresado todos los datos obligatorios.

Nombre: 40. Administrador elimina ítem del buscador.

Descripción: Un agente elimina un registro de ítem del buscador.

Notas:

- Se muestran todos los datos del ítem del buscador.

Escenarios:

40.1. La eliminación del registro de ítem se produce satisfactoriamente.

40.2. No se produce la eliminación por algún error en la base de datos.

Nombre: 41. Usuario público busca inmuebles.

Descripción: Un usuario en Internet entra al sitio Web y consulta los inmuebles disponibles según el tipo de inmueble.

Notas:

- Se muestra la ubicación del inmueble.
- Se muestra la descripción del inmueble.
- Se muestra el valor de arrendamiento del inmueble.

Escenarios:

41.1. La consulta de inmuebles según su tipo se produce satisfactoriamente.

41.2. No se presentan resultados porque no existen datos para mostrar.

2.4. Análisis de la interacción Hombre – Máquina

La Interacción Hombre – Máquina trata acerca del buen diseño de sistemas computacionales que les permita a las personas que los usan desarrollar sus actividades de manera productiva y segura [2]. Es por esta razón que el sistema que estamos desarrollando para esta tesis debe cumplir con varios principios fundamentales de la Interacción Hombre – Máquina, los cuales se mencionan a continuación:

- **Visibilidad:** Todas las opciones y controles del sistema deben ser fácilmente distinguibles para los usuarios, así como el resultado de su uso debe proporcionar la debida retroalimentación, permitiendo que el sistema sea entendible.

- **Permisividad (affordance):** Este principio se refiere a las operaciones y manipulaciones que se pueden hacer con un objeto en particular, esto es que el sistema deberá proporcionar nombres de menús y opciones, así como de controles que tengan concordancia con lo que el usuario espera como resultado de utilizar estos objetos.

- **Usabilidad:** Es importante que el sistema sea fácil de aprender y de usar, especialmente para novatos, con el objetivo de que las tareas de los usuarios sean más eficientes, efectivas y seguras. Por ejemplo,

cuando los usuarios necesiten hacer ingresos de nuevos registros, sólo debe ser necesario llenar los campos que se requieren, enviar la información, y obtener el resultado de la operación en pocos pasos.

- **Consistencia:** El diseño del sistema debe guardar consistencia tanto con los estándares de diseño Web bajo los cuales los usuarios se encuentran familiarizados, así como dentro del mismo sistema, esto es que todos los colores, mensajes de error y resultado, fuentes y tamaños de las letras, enlaces y controles guarden el mismo formato en todas las páginas del sistema.

- **Navegabilidad:** El sistema debe proporcionar que los menús permitan la exploración general de todas las opciones que brinda el sistema a través de hipervínculos fáciles de identificar. Se ha escogido el uso de menús pull-down[7] para presentar las opciones de los menús.

- **Retroalimentación (feedback):** El sistema debe procurar en todo momento indicar al usuario en qué sesión se encuentra, qué acción ha sido ejecutada o se encuentra en estado de ejecución, y cuáles son las operaciones permitidas a las que tiene acceso.

- **Interfaz significativa (familiaridad):** Este principio se relaciona con la importancia de que los usuarios puedan familiarizarse con los obje-

[7] Menús cuyos ítems aparecen cuando se presiona encima del título del menú.

tos y la interface del sistema una vez que ya lo han aprendido a utilizar, proporcionando nombres significativos (fáciles de recordar) de opciones y controles.

- **Seguridad:** Este principio involucra la protección al usuario de condiciones indeseables o peligrosas. Ante esto, el sistema debe proporcionar que el proceso de sincronización de datos sea completo y eficaz.

- **Reconocimiento de errores:** Se deben proporcionar mensajes claros, en lenguaje natural, y fáciles de entender cuando se produzca un error. También se debe dar la facilidad de recuperarse de un error cuando este se produce.

- **Diseño minimalista:** El sistema debe presentar a los usuarios únicamente la información relevante, como el estado del sistema y la operación que se va o está ejecutando.

- **Codificación gráfica:** El uso de los colores dentro del diseño de las páginas Web debe permitir al usuario establecer una separación entre la cabecera, los menús con sus opciones, y la parte central donde se muestran las operaciones y resultados del sistema. Es indispensable evitar el uso excesivo de colores (polución de colores) y la combina-

ción de colores fuertes por la dificultad de interpretar la interface del sistema que esto conlleva.

2.5. Herramientas empleadas en el desarrollo del sistema

Como uno de los objetivos de este proyecto de tesis es facilitar la transmisión y distribución de los contenidos independientemente de la plataforma en la que se han desarrollado, el sistema operativo bajo el cual debe funcionar este proyecto puede ser indistintamente Windows o Linux, debido a la ausencia de componentes instalables, y porque el soporte para páginas dinámicas en PHP está disponible en ambas plataformas, lo cual no afecta el desarrollo del proyecto.

En la misma forma, se necesita un servidor Web que pueda operar indiferentemente en cualquier plataforma y que posea un buen rendimiento. Hemos escogido el servidor Web Apache debido a que éste es un servidor que puede ser utilizado tanto en Windows como en Linux, a diferencia de Internet Information Server (IIS) que sólo opera bajo Windows. Además, Apache no necesita licenciamiento, posee buen rendimiento, alta estabilidad y solidez, y en ambiente Windows el proceso de instalación y configuración es proporcionado por un wizard[8].

[8] El wizard es un programa que asiste en la ejecución de tareas complejas a los usuarios.

En el desarrollo de la aplicación demostrativa de este proyecto de tesis debemos tomar en cuenta el uso de las herramientas de desarrollo que se utilizarán para el diseño de las páginas Web dinámicas, la programación en PHP y la construcción del servicio sincronizador de datos.

Para el desarrollo de las páginas Web dinámicas las alternativas más comúnmente utilizadas en el mercado son: Microsoft FrontPage y Macromedia Dreamweaver. Se ha escogido Dreamweaver por la facilidad de uso de su editor, ya que conjuga las opciones de diseño WYSIWYG[9] (que consiste en diseñar una página Web sin necesidad de escribir ningún código) o a través de código; también porque proporciona soporte a una amplia variedad de tecnologías, entre ellas PHP, y porque el potencial del software en cuanto a la capacidad de programar bajo cualquier lenguaje es de lo más amplio, permitiendo la creación de aplicaciones y diseños Web complejos.

En cuanto al desarrollo de la lógica de programación en PHP, se tienen algunas opciones de IDE[10] OpenSource que se encuentran disponibles en la actualidad, como Zend Studio, NuSphere PhpED, PHP Expert Editor, PHP Designer, CoolEdit, PHP Coder, entre otros. Se ha escogido Zend Studio porque es una herramienta orientada a desarrollar aplicaciones Web en lenguaje PHP, el cual, además de servir de editor de texto para páginas PHP, proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código. Adicionalmente, esta herramienta

[9] "Lo que ve es lo que obtiene" ("what you see is what you get").
[10] Ambiente integrado de desarrollo (Integrated Development Environment), son herramientas que permiten el desarrollo de programas, integrando varias funcionalidades en una sola aplicación

tiene la ventaja de ofrecer versiones para Linux, Windows y MacOS. La desventaja de Zend Studio es que como está escrito en Java, a veces no funciona tan rápido como otras aplicaciones de uso diario.

Finalmente, para el desarrollo del servicio de sincronización de datos se tienen en consideración dos alternativas: la primera, es trabajar con el servicio de replicación y sincronización nativo que proporcionan prácticamente todos los motores de bases de datos. El inconveniente de esta propuesta es que se perdería la facilidad de construir un sistema que utilice cualquier motor de base de datos independiente de donde fue desarrollado inicialmente el sistema. La otra alternativa es desarrollar un programa que se conecte con la aplicación principal y que establezca conexiones con las bases de datos que van a sincronizar, lo que da la ventaja de tener un servicio de sincronización que tan sólo deba adaptarse al motor de base de datos que sea utilizado. Aparece como desventaja la necesidad de establecer una combinación con el sistema operativo para utilizar el servicio de sincronización de datos de forma permanente, pero esto puede ser solucionado utilizando las opciones de "Tareas programadas" en el sistema operativo Windows, y "crontab" en Linux. Por las razones anteriormente expuestas, para este proyecto de tesis se ha decidido desarrollar el servicio de sincronización independientemente de las alternativas que presentan los motores de bases de datos, y para ello se va a utilizar, al igual que en la aplicación, el lenguaje de programación PHP.

CAPITULO

3

Diseño del sistema

CAPÍTULO 3

DISEÑO DEL SISTEMA

3.1. Diseño de la arquitectura del sistema

En esta sección se hace una descripción de la arquitectura del modelo distribuido bajo el cual se fundamenta esta tesis.

En primera instancia hay que describir el servidor de los nodos locales que luego se conecta a otros nodos similares para sincronizar sus datos (Figura 3.1). Los servidores nodos contienen las aplicaciones necesarias que permiten atender los requerimientos de las páginas dinámicas a las que van a acceder los usuarios.

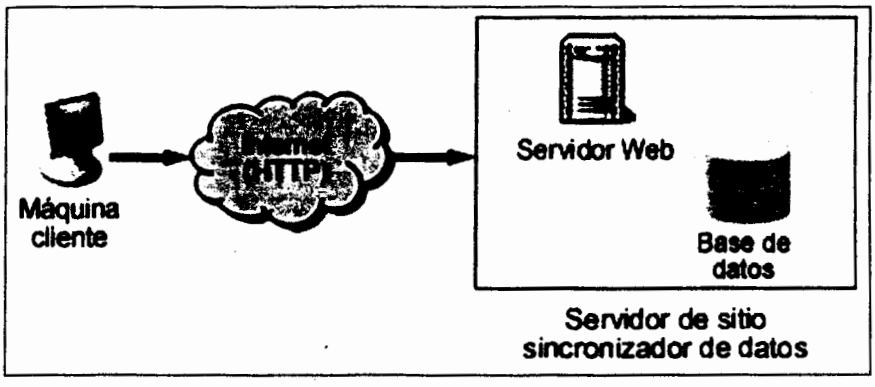


Figura 3.1. Arquitectura de un nodo sincronizador de datos

De igual manera estos servidores poseen los datos con los cuales interactúan los usuarios de una red específica. De esta forma, cuando un usuario cliente desea acceder al sistema a través de un browser[11], hace un requerimiento al servidor Web, el cual transmite la página de inicio del sitio Web, la cual ya contiene información proporcionada por la base de datos del sitio. Si el usuario desea ingresar al sistema con su nombre de usuario y contraseña, envía el pedido al servidor Web el cual verifica los datos y, si éstos son correctos, establece una sesión[12] entre el cliente y el servidor Web.

A partir de este momento el usuario ejecuta las transacciones que requiera, envía el requerimiento al servidor Web, quien lo procesa conectándose a la base de datos y devuelve la respuesta al cliente. Cuando el usuario deja de utilizar el sistema se cierra la sesión establecida entre el cliente y el servidor.

La arquitectura de estos nodos está basada en una arquitectura Cliente – Servidor de múltiples capas, la cual está compuesta por la capa de presentación, la capa de lógica del negocio, y la capa de datos, tal como se muestra en la Figura 3.2. Este diseño es ampliamente aceptado debido a la facilidad de mantenimiento y crecimiento que proporciona.

[11] Aplicación que presenta el contenido de una página Web interpretando un archivo HTML.
[12] Una sesión es una conexión creada para intercambiar datos.

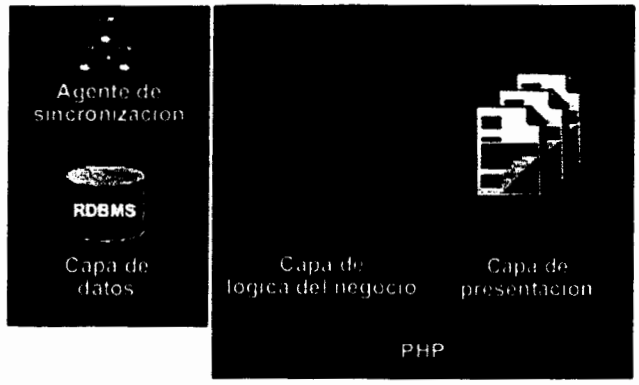


Figura 3.2. Arquitectura Cliente – Servidor

La capa de datos establece la forma mediante la cual se administra el almacenamiento, actualización y consulta de la información, y se permite que la misma sea persistente guardando su integridad. Esta capa está formada por los sistemas administradores de bases de datos relacionales (RDBMS), y por los esquemas de datos que son propios de cada aplicación.

La capa de lógica del negocio determina el comportamiento del sistema, el cual es definido por los componentes que modelan la lógica del negocio de las organizaciones. Estos componentes reciben los requerimientos de la capa de presentación y llevan a cabo las acciones necesarias en la capa de datos para manipular la información del sistema. Está compuesta por los servidores Web y los servidores de aplicación. Este esquema permite una integración más sencilla y eficaz con sistemas externos.

La capa de presentación proporciona todos los elementos que constituyen la interfaz con el usuario, lo cual incluye aquello con lo que el usuario puede

enviar y recibir información desde y hacia la capa de lógica de negocio para su procesamiento, como por ejemplo las pantallas de las aplicaciones, el modelo de navegación del sistema y las aplicaciones para cada modo de acceso (browser, teléfono celular, etc.).

De la misma forma, el modelo bajo el cual se ha considerado crear el servicio de sincronización bajo este sistema distribuido, toma en consideración que cada nodo compuesto por el conjunto de máquinas clientes, servidores y base de datos, debe enlazarse con los otros nodos, tal como se trató en el análisis de tecnologías del capítulo 2, para lo cual debe utilizar el protocolo TCP/IP[13] para la comunicación y el envío de datos entre estos componentes, tal como se muestra en la figura 3.3.

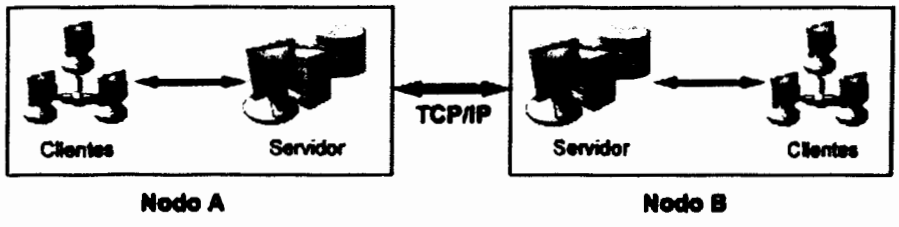


Figura 3.3. Arquitectura del modelo de sincronización

[13] Transport Control Protocol / Internet Protocol (TCP/IP), es un conjunto de protocolos de red que permiten la comunicación de redes interconectadas, e incluye estándares de comunicación.

3.2. Diseño de la base de datos distribuida

3.2.1. Justificación del uso de la base de datos

Para la realización de esta tesis se requiere el desarrollo de un modelo de datos distribuido, en el que se ha considerado dividir el diseño de la base de datos en 2 modelos:

- Modelo de datos de los contenidos.
- Modelo de datos del negocio.

Esta división se ha hecho con el fin de que el modelamiento del sistema permita la independencia en el manejo de la presentación de la información, es decir, la administración de contenidos, con respecto a los datos propios de una organización, lo que permite que el diseño de base de datos propuesto se adapte a cualquier modelo de negocios manteniendo la estructura de los contenidos. Una de las ventajas de este enfoque es que el diseño del modelo de contenidos se adapta a cualquier tipo de tecnología, desde páginas simples hasta información altamente dinámica que se origina de una base de datos que posee un elevado número de transacciones en línea.

En el modelo de datos de los contenidos se almacenan todos los elementos que permiten la administración de la información que da soporte a la presen-

tación de los datos del negocio. Es en este modelo donde se manejan las opciones que posee el sistema con sus elementos de manipulación de la información, así como los usuarios y permisos que restringen el acceso a las opciones del sistema.

El modelo de datos del negocio contiene toda la información propia de las transacciones que se ejecutan dentro de una organización y que tienen importancia en la gestión del negocio, así como en la arquitectura Cliente – Servidor de 3 capas componen la capa de lógica de negocios.

Un aspecto importante para decidir el motor de bases de datos que se va a utilizar en este proyecto, es que éste debe estar disponible para prácticamente cualquier plataforma, tal como se requiere como uno de los objetivos de esta tesis.

Por tanto se ha escogido como herramienta para el almacenamiento de los datos PostgreSQL, el cual es un sistema administrador de bases de datos relacional (RDBMS[14]), open–source, que soporta diferentes funcionalidades, tales como querys[15] complejos, disparadores (triggers[16]), vistas[17], integridad relacional[18] y seguridad de las transacciones [13].

Una ventaja adicional que proporciona el uso de PostgreSQL es que puede ser extendido de muchas maneras, por ejemplo, añadiendo nuevos tipos de

datos, funciones, operadores, funciones agregadas[19], o lenguajes procedurales[20] [13].

Otra ventaja es que debido a su licencia abierta, PostgreSQL puede ser utilizado, modificado y distribuido libremente, y toda la documentación esta disponible gratuitamente. Existen versiones de PostgreSQL para Linux, Windows, Sun Solaris, IBM AIX y FreeBSD, lo que asegura la facilidad de usar los diseños planteados bajo cualquier plataforma.

Por estos motivos se ha considerado el uso de PostgreSQL para el almacenamiento de las tablas y los datos de los modelos de datos de contenidos y negocio del sistema desarrollado en esta tesis.

---

[14] RDBMS (Relational Database Management System) es un sistema que administra datos almacenados en tablas, las cuales se relacionan entre sí.

[15] Los querys son consultas para recuperar datos de una base. Estas consultas son escritas en lenguaje SQL (Structured Query Language), el cual es un lenguaje estándar que permite la comunicación de las aplicaciones con las bases de datos.

[16] El trigger es un código que se ejecuta automáticamente antes o después de una transacción en la base de datos. Si ocurre un evento disparador entonces se llama a una función trigger en el momento apropiado para manejar el evento.

[17] Una vista permite dar un nombre a una consulta (query) para referirse a ésta.

[18] La integridad relacional se refiere al uso apropiado de las claves foráneas que mantienen las relaciones entre las tablas.

[19] Una función agregada permite el cálculo de un resultado simple a partir de un resultado de múltiples filas. Por lo general se usan count, sum (suma), avg (promedio), max (máximo) y min (mínimo).

[20] Un lenguaje procedural permite construir funciones definidas por los usuarios en otros lenguajes que no sean SQL o C. Los lenguajes procedurales disponibles en PostgreSQL son: PL/pgSQL, PL/Tcl, PL/Perl, y PL/Phyton.

3.2.2. Diagramas entidad-relación

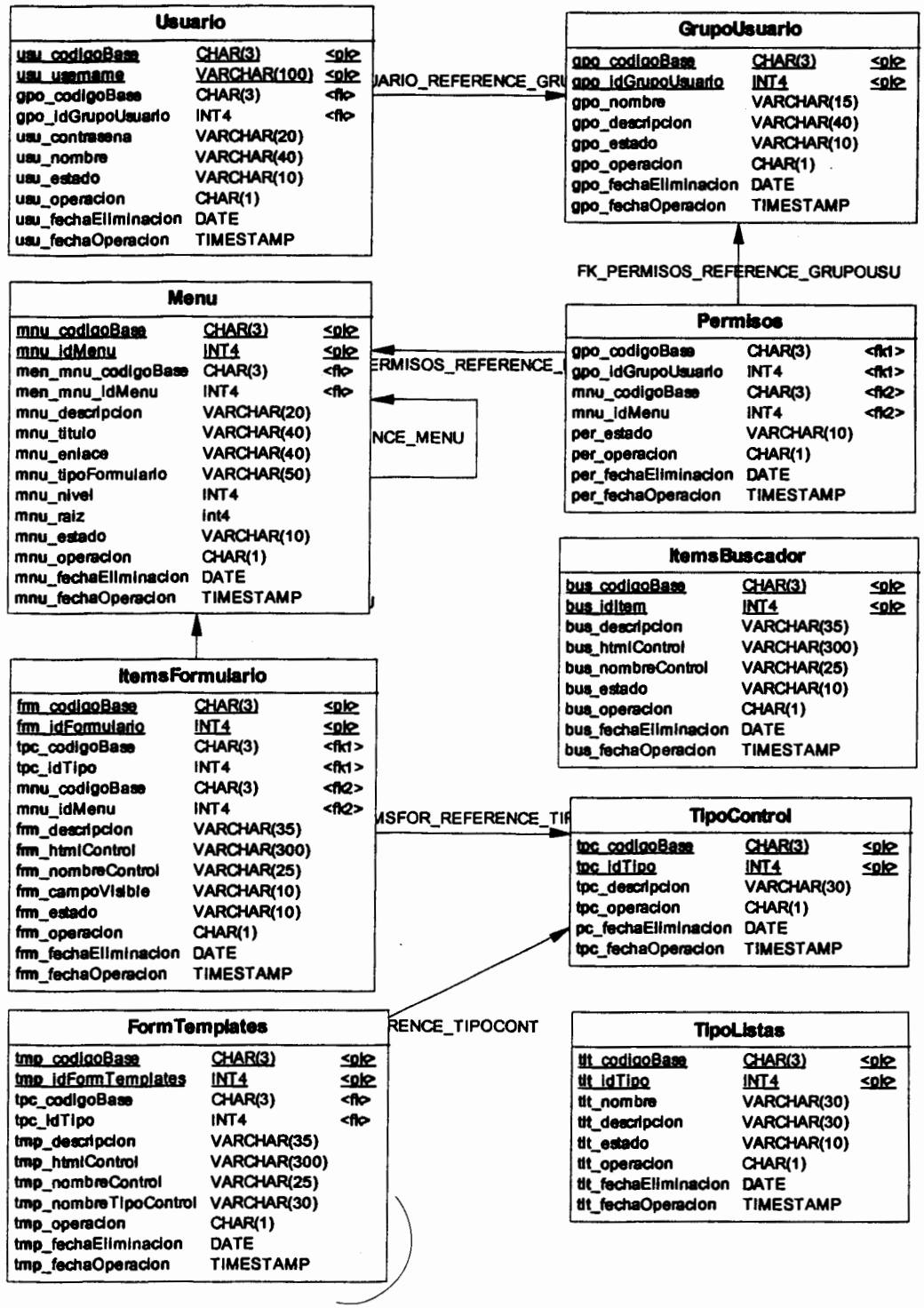


Figura 3.4. Modelo de datos de los contenidos

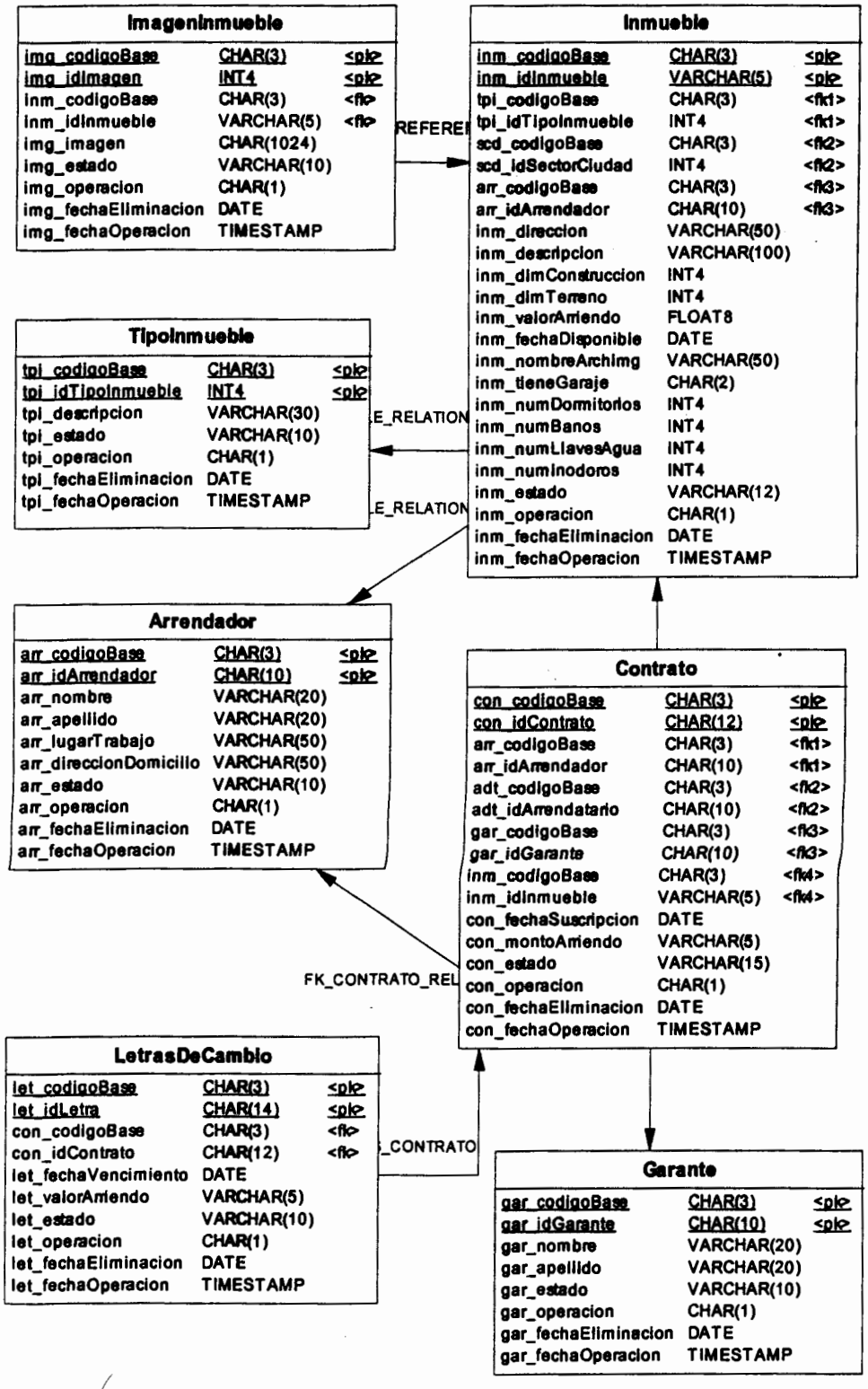


Figura 3.5. Modelo de datos del negocio (Parte I)

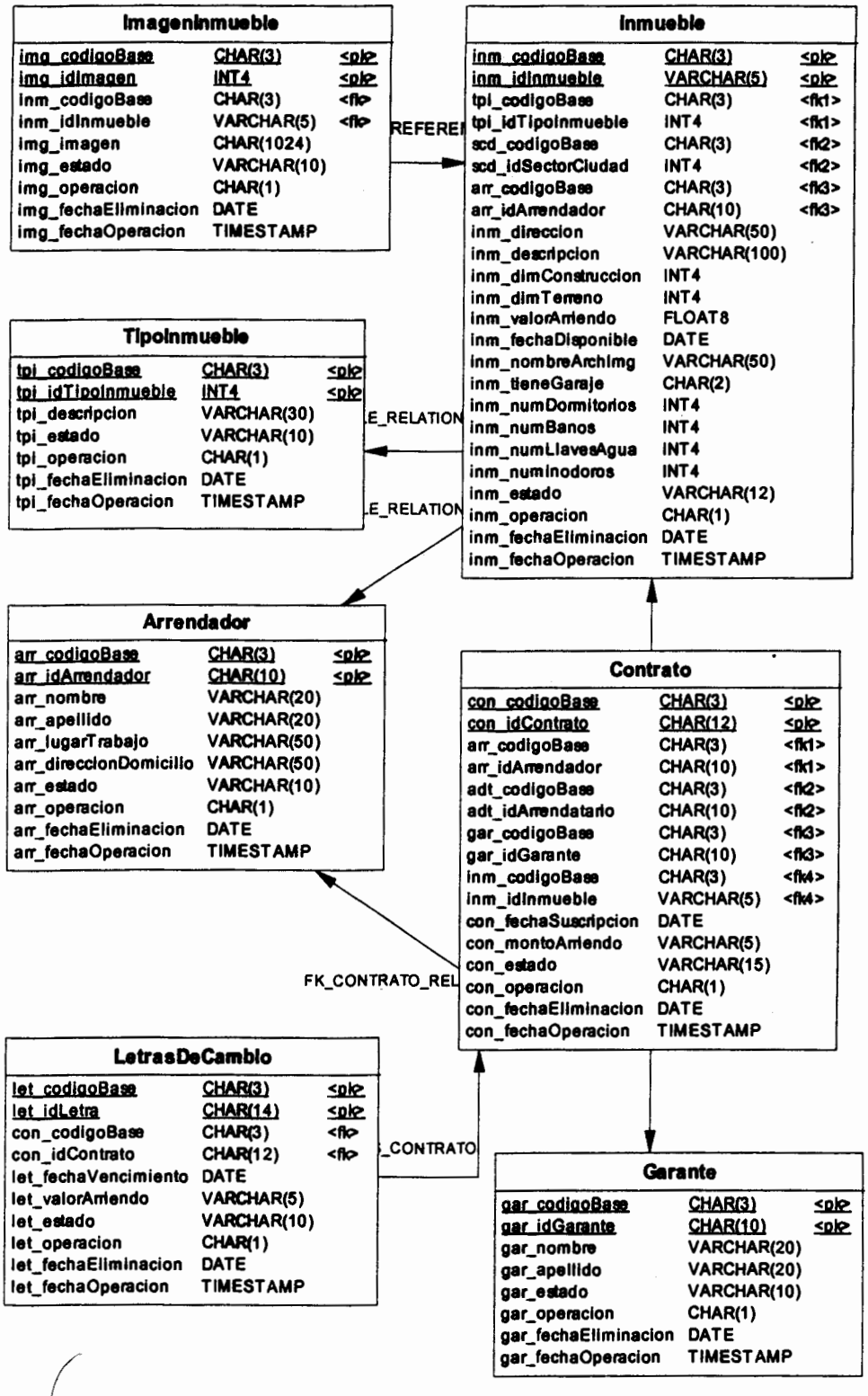


Figura 3.5. Modelo de datos del negocio (Parte I)

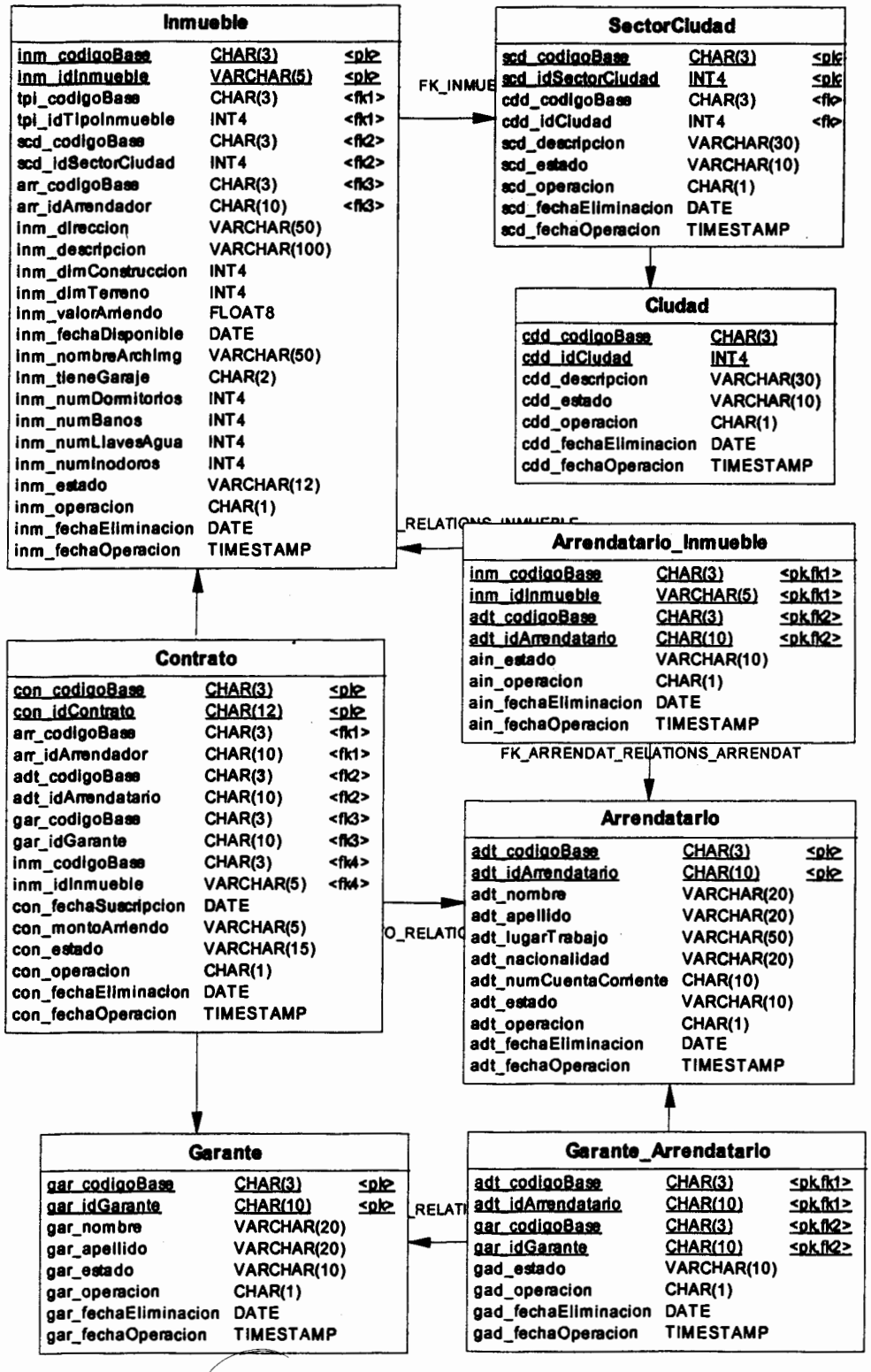


Figura 3.6. Modelo de datos del negocio (Parte II)

3.2.3. Descripción de las entidades y consideraciones de distribución

El modelo de contenidos, tal como se aprecia en la Figura 3.4, posee las siguientes tablas:

- GrupoUsuario.
- Menú.
- Usuario.
- Permisos.
- TipoControl.
- ItemsFormulario.
- ItemsBuscador.
- FormTemplates.
- TipoListas.

La tabla GRUPOUSUARIO es la que almacena los grupos a los que deben pertenecer los usuarios registrados en el sistema. Tiene los siguientes campos:

- gpo_codigoBase: Clave primaria. Campo de tipo Char (caracter) con longitud máxima de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- gpo_idGrupoUsuario: Clave primaria. Campo de tipo Integer (entero), almacena el identificador que corresponde al registro de grupo de usuarios.

- gpo_nombre: Campo de tipo Varchar (caracter variable) con longitud máxima de 15 caracteres, indica el nombre del grupo de usuarios.
- gpo_descripcion: Campo de tipo Varchar con longitud máxima de 40 caracteres, contiene una descripción breve del grupo de usuarios.

- gpo_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el grupo de usuarios está activo o inactivo.
- gpo_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- gpo_fechaEliminacion: Campo de tipo Date (fecha), determina la fecha en la que un registro de grupo de usuarios ha sido eliminado.
- gpo_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción de registro.

La tabla MENU es la que almacena las opciones de menú que presenta las operaciones que debe realizar el sistema. Tiene los siguientes campos:

- mnu_codigoBase: Clave primaria. Campo de tipo Char con longitud máxima de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- mnu_idMenu: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde al registro de menú.

- mnu_descripcion: Campo de tipo Varchar con longitud máxima de 20 caracteres, indica el nombre del menú que va a aparecer en el sistema.

- mnu_enlace: Campo de tipo Varchar con longitud máxima de 40 caracteres, indica el enlace a la página PHP a donde lleva el menú. Este enlace por defecto será el símbolo numeral (#), e indica que el menú tiene opciones de submenú.
- mnu_titulo: Campo de tipo Varchar con longitud máxima de 40 caracteres, indica el título de la operación relacionada con el submenú seleccionado.
- mnu_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el menú está activo o inactivo.
- mnu_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de menú determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- mnu_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de menú ha sido eliminado.
- mnu_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción de registro.
- men_mnu_codigoBase: Clave foránea de la tabla Menú.
- men_mnu_idMenu: Clave foránea de la tabla Menú.
- mnu_nivel: Campo de tipo entero, indica el nivel de profundidad del menú.
- mnu_raiz: Campo de tipo entero, indica el id del menú de nivel cero(raíz) de un menú dado.

- mnu_tipoformulario: Campo de tipo Varchar con longitud máxima de 50 caracteres, indica el tipo de formulario que se presenta para una opción de menú.

La tabla USUARIO es la que almacena los registros de los usuarios registrados en el sistema. Tiene los siguientes campos:

- usu_codigoBase: Clave primaria. Campo de tipo Char con longitud máxima de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- usu_username: Clave primaria. Campo de tipo Varchar con longitud máxima de 20 caracteres, almacena el nombre de usuario con el que una persona ingresa al sistema.
- gpo_codigoBase: Clave foránea de la tabla GrupoUsuario.
- gpo_idGrupoUsuario: Clave foránea de la tabla GrupoUsuario.
- usu_contrasena: Campo de tipo Varchar con longitud máxima de 100 caracteres, almacena la contraseña del usuario.
- usu_nombre: Campo de tipo Varchar con longitud máxima de 40 caracteres, contiene el nombre completo del usuario (nombres y apellidos).
- usu_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el usuario está activo o inactivo.
- usu_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de usuario de-

terminado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).

- usu_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de usuario ha sido eliminado.
- usu_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción de registro.

La tabla PERMISOS es una tabla transaccional que resulta de la relación entre las tablas Menú y GrupoUsuario, y que indica los permisos de las opciones a los que tiene acceso un usuario, según su grupo. Tiene los siguientes campos:

- gpo_codigoBase: Clave foránea de la tabla GrupoUsuario.
- gpo_idGrupoUsuario: Clave foránea de la tabla GrupoUsuario.
- mnu_codigoBase: Clave foránea de la tabla Menú.
- mnu_idMenu: Clave foránea de la tabla Menú. Indica los menús a los que tiene acceso el grupo de usuarios.
- per_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- per_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el permiso está activo o inactivo.
- per_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

- per_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de permiso ha sido eliminado.

La tabla TIPOCONTROL es la que registra todos los tipos de controles Web que aceptará el sistema. Tiene los siguientes campos:

- tpc_codigoBase: Clave primaria. Campo de tipo Char con longitud máxima de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- tpc_idTipo: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde al registro de un tipo de control.

- tpc_descripcion: Campo de tipo Varchar con longitud máxima de 30 caracteres, indica el nombre del control Web.
- tpc_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de tipo de control determinado. Los valores que puede tomar son 'I' o 'U'.
- tpc_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.
- tpc_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de tipo de control ha sido eliminado.

La tabla ITEMSFORMULARIO es la que almacena todos los controles y opciones de un formulario seleccionado por un usuario, y la disposición en la que se van a presentar en el mismo. Tiene los siguientes campos:

- frm_codigoBase: Clave primaria. Campo de tipo Char con longitud máxima de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- frm_idFormulario: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde al registro de un control.
- tpc_codigoBase: Clave foránea de la tabla TipoControl.
- tpc_idTipo: Clave foránea de la tabla TipoControl. Indica el tipo de control Web al que hace referencia el registro.
- frm_descripcion: Campo de tipo Varchar con longitud máxima de 35 caracteres, indica la descripción que va a aparecer en el formulario junto a un control determinado.
- frm_htmlControl: Campo de tipo Varchar con longitud máxima de 300 caracteres, indica el código HTML para un control Web.
- frm_nombreControl: Campo de tipo Varchar con longitud máxima de 25 caracteres, indica el nombre del control Web.
- frm_campoVisible: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el control Web debe aparecer al inicio o al final dentro de un formulario. Al inicio indica que sólo aparecerá cuando antes de ingresar a un formulario principal, se debe ingresar algún código de identificación para obtener los datos de un registro determinado. Al final indica que sólo aparecerá en un formulario principal que determinará el ingreso, actualización o eliminación de un registro de la base de datos.

- frm_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el control Web está activo o inactivo.
- frm_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de ítem de formulario determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- frm_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de ítem de formulario ha sido eliminado.
- frm_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.
- mnu_codigoBase: Clave foránea de la tabla Menú.
- mnu_idMenu: Clave foránea de la tabla Menú. Indica los menús a los que tiene acceso el grupo de usuarios.

La tabla ITEMSBUSCADOR es la que almacena los controles (elementos) que va a poseer el buscador de inmuebles del sistema. Tiene los siguientes campos:

- bus_codigoBase: Clave primaria. Campo de tipo Char con longitud máxima de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- bus_idItem: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde al registro de un control del buscador.

- bus_descripcion: Campo de tipo Varchar con longitud máxima de 35 caracteres, indica la descripción que va a aparecer en el buscador junto a un control determinado.
- bus_htmlControl: Campo de tipo Varchar con longitud máxima de 300 caracteres, indica el código HTML para un control del buscador.
- bus_nombreControl: Campo de tipo Varchar con longitud máxima de 25 caracteres, indica el nombre del control Web del buscador.
- bus_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el control del buscador está activo o inactivo.
- bus_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de ítem de buscador determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- bus_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de ítem de buscador ha sido eliminado.
- bus_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla FORMTEMPLATES es la que tabla que contiene los controles que sirven como plantillas para generar los formularios (contenido) de los submenús. Tiene los siguientes campos:

- tmp_codigoBase: Clave primaria. Campo de tipo Char con longitud máxima de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- tmp_idFormTemplates: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde al registro de un elemento del formulario plantilla.
- tpc_codigoBase: Clave foránea de la tabla TipoControl.
- tpc_idTipo: Clave foránea de la tabla TipoControl. Indica el tipo de control Web que va a ser ingresado dentro de un formulario.
- tmp_descripcion: Campo de tipo Varchar con longitud máxima de 35 caracteres, indica la descripción que va a aparecer en el formulario junto a un control determinado.
- tmp_htmlControl: Campo de tipo Varchar con longitud máxima de 300 caracteres, indica el código HTML para el control.
- tmp_nombreControl: Campo de tipo Varchar con longitud máxima de 25 caracteres, indica el nombre del control Web.
- tmp_nombreTipoControl: Campo de tipo Varchar con longitud máxima de 30 caracteres, indica el tipo del control Web que se va a presentar en el formulario plantilla.
- tmp_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de un ítem del formulario plantilla. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).

- tmp_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.
- tmp_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de ítem de buscador ha sido eliminado.

La tabla TIPOLISTAS es la que almacena todos los listados (cuyos registros son presentados en un listado de tipo combo) que puede presentar el sistema. Tiene los siguientes campos:

- tlt_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- tlt_idTipo: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde al registro de un tipo de lista.
- tlt_nombre: Campo de tipo Varchar con longitud máxima de 30 caracteres, indica el nombre del control tipo lista (cbonombrecontrol).
- tlt_descripcion: Campo de tipo Varchar con longitud máxima de 30 caracteres, describe a qué lista se refiere el control escogido.
- tlt_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el registro del tipo de lista está activo o inactivo.
- tlt_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de tipo de lista determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).

- tlt_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de tipo de lista ha sido eliminado.
- tlt_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

El modelo de negocio, tal como se aprecia en las Figuras 3.5 y 3.6, posee las siguientes tablas:

- Arrendador.
- Arrendatario.
- Garante.
- Ciudad.
- SectorCiudad.
- TipoInmueble.
- Inmueble.
- ImagenInmueble.
- Contrato.
- LetrasDeCambio.
- Garante_Arrendatario.
- Arrendatario_Inmueble.

La tabla ARRENDADOR es la que almacena los registros de los arrendadores propietarios de los inmuebles registrados en el sistema. Tiene los siguientes campos:

- arr_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- arr_idArrendador: Clave primaria. Campo de tipo Char con longitud de 10 caracteres, almacena el número de cédula o pasaporte del arrendador.
- arr_nombre: Campo de tipo Varchar con longitud máxima de 20 caracteres, indica el(los) nombre(s) del arrendador.
- arr_apellido: Campo de tipo Varchar con longitud máxima de 20 caracteres, indica el(los) apellido(s) del arrendador.
- arr_lugarTrabajo: Campo de tipo Varchar con longitud máxima de 50 caracteres, indica el lugar en que labora el arrendador. No es obligatorio.
- arr_direccionDomicilio: Campo de tipo Varchar con longitud máxima de 50 caracteres, indica la dirección del domicilio del arrendador.
- arr_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el registro del arrendador está activo o inactivo.
- arr_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de arrendador determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- arr_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de arrendador ha sido eliminado.
- arr_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla ARRENDATARIO es la que almacena los registros de los arrendatarios o inquilinos de los inmuebles registrados en el sistema. Tiene los siguientes campos:

- adt_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- adt_idArrendador: Clave primaria. Campo de tipo Char con longitud de 10 caracteres, almacena el número de cédula o pasaporte del arrendatario.
- adt_nombre: Campo de tipo Varchar con longitud máxima de 20 caracteres, indica el(los) nombre(s) del arrendatario.
- adt_apellido: Campo de tipo Varchar con longitud máxima de 20 caracteres, indica el(los) apellido(s) del arrendatario.
- adt_lugarTrabajo: Campo de tipo Varchar con longitud máxima de 50 caracteres, indica el lugar en que labora el arrendatario.
- adt_nacionalidad: Campo de tipo Varchar con longitud máxima de 20 caracteres, indica la nacionalidad del arrendatario.
- adt_numCuentaCorriente: Campo de tipo Char con longitud de 10 caracteres, indica el número de cuenta corriente del arrendatario.
- adt_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el registro del arrendatario está activo o inactivo.
- adt_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de arrendatario

determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).

- adt_fechaEliminacion: Campo de tipo Date determina la fecha en la que un registro de arrendatario ha sido eliminado.
- adt_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla GARANTE es la que almacena los registros de los garantes de los arrendatarios en los contratos de arrendamiento de los inmuebles. Tiene los siguientes campos:

- gar_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- gar_idGarante: Clave primaria. Campo de tipo Char con longitud de 10 caracteres, almacena el número de cédula o pasaporte del garante del contrato.
- gar_nombre: Campo de tipo Varchar con longitud máxima de 20 caracteres, indica el(los) nombre(s) del garante.
- gar_apellido: Campo de tipo Varchar con longitud máxima de 20 caracteres, indica el(los) apellido(s) del garante.
- gar_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el registro del garante está activo o inactivo.

- gar_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de garante determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- gar_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de garante ha sido eliminado.
- gar_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla CIUDAD es la que posee los registros de las ciudades en donde se encuentran los inmuebles. Tiene los siguientes campos:

- cdd_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- cdd_idCiudad: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde al registro de una ciudad.
- cdd_descripcion: Campo de tipo Varchar con longitud máxima de 30 caracteres, se refiere al nombre de la ciudad.
- cdd_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el registro de la ciudad está activo o inactivo.
- cdd_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de ciudad de-

terminado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).

- cdd_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de ciudad ha sido eliminado.
- cdd_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla SECTORCIUDAD es la que posee los registros de los sectores que pertenecen a las ciudades. Tiene los siguientes campos:

- scd_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- scd_idSectorCiudad: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde al registro de un sector de una ciudad.
- cdd_codigoBase: Clave foránea de la tabla Ciudad.
- cdd_idCiudad: Clave foránea de la tabla Ciudad. Indica la ciudad a la que pertenece el registro.

- scd_descripcion: Campo de tipo Varchar con longitud máxima de 30 caracteres, se refiere al nombre del sector de la ciudad.
- scd_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el registro del sector de ciudad está activo o inactivo.
- scd_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de sector de

ciudad determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).

- scd_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de sector de ciudad ha sido eliminado.
- scd_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla TIPOINMUEBLE es la que almacena los registros de las categorías en las que se encuentran clasificados los inmuebles registrados en el sistema. Tiene los siguientes campos:

- tpi_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- tpi_idTipoInmueble: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde al registro de una categoría de inmueble.
- tpi_descripcion: Campo de tipo Varchar con longitud máxima de 30 caracteres, se refiere al nombre del tipo de inmueble.
- tpi_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el registro del tipo de inmueble está activo o inactivo.
- tpi_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de tipo de inmueble determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).

- tpi_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de sector de tipo de inmueble ha sido eliminado.
- tpi_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla INMUEBLE es la que almacena los registros de los inmuebles (casas, departamentos, etc.) del sistema. Tiene los siguientes campos:

- inm_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base donde fue creado el registro.
- inm_idInmueble: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde al registro de inmueble.
- tpi_codigoBase: Clave foránea de la tabla TipoInmueble.
- tpi_idTipoInmueble: Clave foránea de la tabla TipoInmueble. Indica el tipo de inmueble al que pertenece al registro.
- scd_codigoBase: Clave foránea de la tabla SectorCiudad.
- scd_idSectorCiudad: Clave foránea de la tabla SectorCiudad. Indica el sector de una ciudad en donde se encuentra el registro.
- arr_codigoBase: Clave foránea de la tabla Arrendador.
- arr_idArrendador: Clave foránea de la tabla Arrendador. Hace referencia al propietario del inmueble.
- inm_direccion: Campo de tipo Varchar con longitud máxima de 50 caracteres, almacena la dirección (calles, ciudadelas, manzana, etc.) en la cual se encuentra el inmueble.

- inm_descripcion: Campo de tipo Varchar con longitud máxima de 100 caracteres, posee una descripción breve del inmueble.
- inm_dimConstruccion: Campo de tipo Integer, indica las dimensiones de construcción del inmueble establecido en metros cuadrados (m^2).
- inm_dimTerreno: Campo de tipo Integer, indica las dimensiones del terreno del inmueble establecido en metros cuadrados (m^2).
- inm_valorArriendo: Campo de tipo Float, especifica el valor del arriendo mensual que tiene el inmueble.
- inm_fechaDisponible: Campo de tipo Date, indica la fecha desde la cual el inmueble se encuentra disponible para ser alquilado, ya sea porque es un registro nuevo dentro del sistema o porque ya terminó el contrato de arrendamiento sobre dicho inmueble.
- inm_nombreArchImg: Campo de tipo Varchar con longitud máxima de 50 caracteres, indica el nombre del archivo relacionado con el inmueble. Este archivo debe ser una imagen en formato GIF, JPEG o BMP.
- inm_tieneGaraje: Campo de tipo Char con longitud de 2 caracteres, indica si el inmueble posee garaje o no. Sus valores son "Si" o "No".
- inm_numDormitorios: Campo de tipo Integer, indica la cantidad de habitaciones para dormitorios que posee el inmueble. Por defecto dicho valor será '0'.
- inm_numBanos: Campo de tipo Integer, indica la cantidad de baños o instalaciones sanitarias que posee el inmueble. Por defecto dicho valor será '0'.

- inm_numLlavesAgua: Campo de tipo Integer, indica la cantidad de llaves de agua (como lavaderos o lavabos) que posee el inmueble. Por defecto dicho valor será '0'.
- inm_numInodoros: Campo de tipo Integer, indica la cantidad de inodoros que posee el inmueble. Por defecto dicho valor será '0'.
- inm_estado: Campo de tipo Varchar con longitud máxima de 12 caracteres, indica si el registro de inmueble es "Alquilado", "No alquilado" o "Inactivo".
- inm_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de inmueble determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- inm_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de inmueble ha sido eliminado.
- inm_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla IMAGENINMUEBLE es la que almacena la información de las imágenes de los inmuebles en palabras o segmentos de 1024 bytes para que pueda ser transmitida, a través de la sincronización, a los otros repositorios de datos, lo que significa que una imagen podrá estar segmentada en varios registros de esta tabla. Tiene los siguientes campos:

- img_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base donde fue creado el registro.
- img_idImagen: Clave primaria. Campo de tipo Integer, almacena el identificador que corresponde a un segmento de la imagen de un inmueble.
- inm_codigoBase: Clave foránea de la tabla Inmueble.
- inm_idInmueble: Clave foránea de la tabla Inmueble. Hace referencia al inmueble al cual pertenece el segmento de la imagen.
- img_imagen: Campo de tipo Char con longitud de 1024 caracteres, contiene una palabra (segmento de archivo) en que se halla dividida una imagen de un inmueble.
- img_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el registro de imagen de inmueble está activo o inactivo.
- img_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de imagen de inmueble determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- img_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de imagen de inmueble ha sido eliminado.
- img_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla CONTRATO es la que almacena los registros de los contratos celebrados entre un arrendador y un arrendatario para el arrendamiento de un

inmueble durante 1 año; los contratos cuentan con la presencia de un garante. Tiene los siguientes campos:

- con_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base donde fue creado el registro.
- con_idContrato: Clave primaria. Campo de tipo Char con longitud de 12 caracteres, indica el número del contrato.
- arr_codigoBase: Clave foránea de la tabla Arrendador.
- arr_idArrendador: Clave foránea de la tabla Arrendador. Hace referencia al propietario del inmueble.
- adt_codigoBase: Clave foránea de la tabla Arrendatario.
- adt_idArrendatario: Clave foránea de la tabla Arrendatario. Hace referencia al inquilino del inmueble.
- gar_codigoBase: Clave foránea de la tabla Garante.
- gar_idGarante: Clave foránea de la tabla Garante. Hace referencia al garante del contrato.
- inm_codigoBase: Clave foránea de la tabla Inmueble.
- inm_idInmueble: Clave foránea de la tabla Inmueble. Hace referencia al inmueble que está siendo alquilado.
- con_fechaSuscripcion: Campo de tipo Date, indica la fecha en la cual se está celebrando el contrato de arrendamiento de un inmueble entre un arrendador y un arrendatario.
- con_montoArriendo: Campo de tipo Varchar con longitud máxima de 5 caracteres, indica el valor mensual que debe pagar el arrendatario por

concepto del alquiler del inmueble. Este valor debe estar dado en dólares estadounidenses (USD$).

- con_estado: Campo de tipo Varchar con longitud máxima de 15 caracteres, indica si el contrato se encuentra en estado "Activo", "Finalizado" o "Cancelado". El estado "Activo" se produce cuando se crea un nuevo contrato. El estado "Finalizado" se produce cuando el contrato llega a su fin después de 1 año a partir de su fecha de suscripción. El estado "Cancelado" indica que el contrato ha sido concluido antes de su fecha de culminación.
- con_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de contrato determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- con_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de contrato ha sido eliminado.
- con_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla LETRASDECAMBIO es la que almacena los registros de las letras de cambio que corresponden a cada uno de los pagos mensuales que deben realizar los arrendatarios de los inmuebles. Tiene los siguientes campos:

- let_codigoBase: Clave primaria. Campo de tipo Char con longitud de 3 caracteres, almacena el código de la base de datos donde fue creado el registro.
- let_idLetra: Clave primaria. Campo de tipo Char con longitud de 14 caracteres, indica el número de la letra de cambio mensual. Este número se origina del número del contrato más dos dígitos adicionales (del '01' al '12') que especifican cada una de las letras de cambio mensuales que pertenecen a un contrato determinado.
- con_codigoBase: Clave foránea de la tabla Contrato.
- con_idContrato: Clave foránea de la tabla Contrato. Hace referencia al contrato al cual pertenece la letra de cambio.
- let_fechaVencimiento: Campo de tipo Date, indica la fecha en la cual debe ser cancelada la letra de cambio. Esta fecha se origina a partir de la fecha de suscripción del contrato.
- let_valorArriendo: Campo de tipo Varchar con longitud máxima de 5 caracteres, indica el valor mensual que debe pagar el arrendatario por concepto del alquiler del inmueble. Este valor debe estar dado en dólares estadounidenses (US$).
- let_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si la letra de cambio se encuentra en estado "Cancelada", "Pendiente" o "Eliminada". El estado "Cancelada" indica que ya se realizó el pago del arriendo mensual. El estado "Pendiente" se produce cuando se generan las letras de cambio que pertenecen a un contrato y éstas

aún no han sido canceladas. El estado "Eliminada" indica que el contrato ha sido concluido antes de su fecha de culminación, y por lo tanto las letras de cambio también quedan eliminadas.

- let_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de letra de cambio determinado. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- let_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de letra de cambio ha sido eliminado.
- let_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla GARANTE_ARRENDATARIO es una tabla transaccional que resulta de la relación entre las tablas Garante y Arrendatario, y que indica quiénes son los garantes que están relacionados con los arrendatarios de los inmuebles. Tiene los siguientes campos:

- adt_codigoBase: Clave foránea de la tabla Arrendatario.
- adt_idArrendatario: Clave foránea de la tabla Arrendatario. Indica el arrendatario que tiene relación con un garante.
- gar_codigoBase: Clave foránea de la tabla Garante.
- gar_idGarante: Clave foránea de la tabla Garante. Indica el garante que tiene relación con un arrendatario.

- gad_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el registro de esta tabla se encuentra activo o inactivo.
- gad_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de esta tabla. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- gad_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de esta tabla ha sido eliminado.
- gad_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

La tabla ARRENDATARIO_INMUEBLE es una tabla transaccional que resulta de la relación entre las tablas Arrendatario e Inmueble, y que registra un histórico de los arrendatarios que han alquilado los inmuebles. Tiene los siguientes campos:

- inm_codigoBase: Clave foránea de la tabla Inmueble.
- inm_idInmueble: Clave foránea de la tabla Inmueble. Hace referencia al inmueble alquilado por un arrendatario.
- adt_codigoBase: Clave foránea de la tabla Arrendatario.
- adt_idArrendatario: Clave foránea de la tabla Arrendatario. Indica el arrendatario que ha alquilado un inmueble.
- ain_estado: Campo de tipo Varchar con longitud máxima de 10 caracteres, indica si el registro de esta tabla se encuentra activo o inactivo.

- ain_operacion: Campo de tipo Char con longitud de 1 caracter, indica el tipo de operación recientemente realizada en un registro de esta tabla. Los valores que puede tomar son 'I' (inserción) o 'U' (actualización).
- ain_fechaEliminacion: Campo de tipo Date, determina la fecha en la que un registro de esta tabla ha sido eliminado.
- ain_fechaOperacion: Campo de tipo Date, determina la fecha y hora en la que se ha realizado una transacción en un registro.

Adicionalmente, junto a estos modelos, se debe tomar en consideración las siguientes tablas del sistema, las cuales ayudan a efectuar el proceso de sincronización de manera óptima y parametrizable:

- Sys_tablas.
- Sys_log_sincroniza.

La tabla SYS_TABLAS es la que almacena los registros de las tablas registrados en el sistema que pueden sincronizar sus datos hacia otras bases. Tiene los siguientes campos:

- nombre_tabla: Campo de tipo Varchar con longitud máxima de 30 caracteres, indica el nombre de la tabla. El nombre de la tabla se almacena en letras minúsculas sin espacios.
- campos: Campo de tipo Varchar con longitud máxima de 1000 caracteres, indica los campos que posee una tabla. El administrador deberá in-

gresar cada campo sin espacios, con letra minúscula, seguido de punto y coma. Ejemplo: campo1;campo2;campo3.

- campo_comparador: Campo de tipo Varchar con longitud máxima de 100 caracteres, indica el(los) campo(s) que identifican un registro de una tabla y lo diferencian del resto. El administrador deberá ingresar cada campo comparador sin espacios, con letra minúscula, seguido de punto y coma. Ejemplo: comparador1;comparador2;comparador3.

- tipo_operacion: Campo de tipo Varchar con longitud máxima de 50 caracteres, indica el nombre del campo que señala la transacción realizada en un registro de una tabla (Inserción o Actualización).

- fecha_operacion: Campo de tipo Varchar con longitud máxima de 50 caracteres, indica el nombre del campo que señala la fecha en que se realiza la transacción (Inserción o Actualización) en un registro de una tabla.
- fecha_eliminacion: Campo de tipo Varchar con longitud máxima de 50 caracteres, indica el nombre del campo que señala la fecha en que un registro es eliminado del sistema.
- tipo_prioridad: Campo de tipo Char con longitud 1 caracter, indica la prioridad de realizar la transacción cuando existe un conflicto de sincronización. Su valor es 'P' para indicar que la prioridad es para la primera transacción realizada y 'U' para indicar que la prioridad es para la última transacción realizada.

- nivel_prioridad: Campo de tipo Integer, indica el nivel de prioridad de una tabla al momento de efectuar la sincronización.

La tabla SYS_LOG_SINCRONIZA es la que almacena un registro de error por cada transacción que no pudo ser sincronizada. Tiene los siguientes campos:

- nombre_tabla: Campo de tipo Varchar con longitud máxima de 30 caracteres, indica el nombre de la tabla. El nombre de la tabla se almacena en letras minúsculas sin espacios.
- trama_registros: Campo de tipo Varchar con longitud máxima de 5000 caracteres, contiene los valores de los campos de un registro que no se sincronizó. El formato de este campo contendrá cada valor de campo sin espacios, con letra minúscula, seguido de punto y coma. Ejemplo: valor1;valor2;valor3.
- fecha_transaccion: Campo de tipo TimeStamp, indica la fecha y hora en que se produce el error de sincronización.
- tipo_operacion: Campo de tipo Varchar con longitud máxima de 20 caracteres, indica si la transacción que falló era de inserción o actualización. Sus valores pueden ser 'OPERACIÓN INSERT' u 'OPERACIÓN UPDATE', dependiendo el caso.
- destino_fallido: Campo de tipo Varchar con longitud máxima de 200 caracteres, indica la base de datos hacia donde iba dirigido el registro de sincronización fallida. El formato incluye 'host', 'puerto', 'nombre_base', 'nombre_usuario' y 'contraseña'.

Seguidamente, tenemos que tomar en cuenta ciertas consideraciones de distribución al momento de construir los modelos de datos de este proyecto de tesis, las mismas que se señalan a continuación:

En primer lugar, es necesario diseñar las tablas que van a sincronizar de tal modo que al momento de que los datos tengan que pasar de un repositorio a otro no surjan conflictos de replicación. Esto podría ocurrir, por ejemplo, cuando en uno de los repositorios se ingrese un registro de arrendador con identificación '0951264161', y antes de que los datos sincronicen, en otro repositorio también se produzca el mismo ingreso de un registro de arrendador. Esto ocasionaría un fallo en la transferencia de datos cuando se produzca la sincronización porque no se podría copiar el nuevo registro de la base origen a la base destino. Para evitar este conflicto se ha adoptado que las tablas que vayan a sincronizar (que en el caso de este proyecto de tesis se ha considerado que sean todas las tablas, tanto del modelo de datos de los contenidos como del modelo de datos del negocio, sincronicen sus datos) posean una clave primaria compuesta[21], la cual estará formada por el identificador propio de un registro de la tabla y un código de la base de datos de origen.

Este código estará compuesto de 3 caracteres con el siguiente formato: 'TE#', donde # será un dígito del 1 al 9, diferente para cada repositorio de datos, lo que permitirá indicar e identificar en qué base fue creado el registro.

Además, las tablas que necesiten un identificador que en un modelo de datos no distribuido debiera ser secuencial, en este modelo tendrán que tener un identificador de tipo entero, es decir, que no se utilizará el tipo de dato secuencial.

Adicionalmente, se han agregado 4 campos a las tablas del modelo distribuido:

- Estado: Indica el estatus de un registro el cual está especificado en el diccionario de datos dependiendo de cada tabla.

- Operación: Indica si se ha registrado alguna operación en el registro, lo cual permitirá al sincronizador determinar si este registro tiene que ser enviado a los otros repositorios de datos. Si el registro ha sido creado y aún no ha sido replicado, su campo operación será marcado con 'I' (de "Insert"). Si el registro ha sido modificado y aún no ha sido replicado, su campo operación será marcado con 'U' (de "Update"). Una vez que el sincronizador haya enviado los datos y actualizado las tablas de los otros repositorios de datos, el valor del campo operación de los registros marcados será cambiado a "Nulo" (null), lo cual indicará que el registro ya ha sido sincronizado. Cabe recalcar, que en el caso que no se lograse completar la sincronización de algún de registro, éste será almacenado en el log de errores de sincronización, para posterior manipulación manual.

- Fecha de operación: Indica la fecha y hora en que se ha efectuado una transacción en la base de datos (Insert o Update). Este campo es fundamental como ayuda en el proceso de sincronización para resolver conflictos. Su relevancia radica en el hecho de que cuando se produce una transacción en el mismo registro, pero en bases diferentes, la transacción que finalmente se ejecuta, depende del nivel de prioridad establecido en el campo "tipo_prioridad" de la tabla Sys_tablas. Si la prioridad es para la primera transacción, el registro con la fecha de operación más antigua, es el que se sincronizará a las otras bases, mientras que el registro fallido, se almacenará en el log de errores de la tabla Sys_log_sincroniza. Por el contrario, si la prioridad es para la última transacción, el registro con la fecha de operación más reciente es el que se sincronizará a las otras bases.

- Fecha de eliminación: Indica la fecha en que un registro ha sido eliminado de la base de datos. Es preciso anotar que, al tratarse de un modelo de datos distribuido, no se puede aplicar el borrado físico de los registros de la base de datos, por lo que es conveniente aplicar la eliminación lógica de los registros. La eliminación lógica de un registro significa que cuando un usuario que utiliza el sistema elimina un registro, por ejemplo de arrendatario, éste registro no desaparece físicamente de la base de datos sino que es marcado como "borrado"; para esta transacción se actualiza el valor del campo 'Estado' de cada ta-

bla como "Inactivo" o "Eliminado" (según el criterio especificado anteriormente en la descripción de las entidades de la base de datos distribuida), se marca el campo 'Operación' con 'U' (de "Update") y se registra en el campo 'Fecha de Eliminación' la fecha en que se ha producido la transacción. El hecho de establecer el campo 'Fecha de Eliminación' permitirá que en un momento determinado una aplicación adicional del sistema elimine físicamente de la base de datos los registros cuya fecha de eliminación exceda cierto tiempo que ya hace inútil su conservación dentro de las tablas de la base.

Finalmente, se ha creado una tabla que va a contener la información de las imágenes relacionadas a los inmuebles registrados en el sistema, debido a que cuando un usuario adjunta una imagen en un registro de inmueble, ésta tiene que ser replicada a los otros repositorios de datos, lo cual se debe realizar a través del servicio de sincronización para que la imagen sea transferida a los otros servidores del modelo distribuido.

3.3. Diseño del sincronizador de datos

El sincronizador de datos deberá tener en cuenta dos aspectos para su funcionamiento: se deberá ejecutar constantemente como un proceso en segundo plano dentro del sistema operativo donde se encuentra el servidor de

la base de datos, y deberá establecer un mecanismo de comunicación con los otros repositorios de datos con los que va a sincronizar cada uno de los registros de las tablas que tengan marcado el campo 'Operación', tal como se definió en las consideraciones de distribución.

Para asegurar que la ejecución del sincronizador se realice constantemente se ha determinado utilizar las herramientas propias que proporcionan los sistemas operativos. En el caso de Windows, se ha considerado usar la herramienta "Tareas programadas"[22], la cual permite programar comandos, documentos o programas para que sean ejecutados por el sistema operativo en un momento previamente determinado o en intervalos de tiempo. El mismo procedimiento se establece en Linux, en donde se debe utilizar el comando "crontab" que permite programar la ejecución de algún programa o documento en un momento determinado. En este caso se va a programar la ejecución de un script que es el programa que va a realizar la sincronización de datos entre los diferentes puntos del sistema. El programa mencionado anteriormente será desarrollado en PHP, tal como se señaló en el análisis del sistema, y en primera instancia se conectará con los otros repositorios de datos a través de la función pg_pconnect de PHP. Esta función permite abrir una conexión persistente a una base de datos en PostgreSQL y establecer este recurso para las otras funciones que se conectarán a la base de datos. Los argumentos que están disponibles en pg_pconnect incluyen host, port, tty, options, dbname, user, y password [12].

[22] En Windows existe el comando "at" para programar un comando, una secuencia de ellos o un programa de forma que se ejecute en una hora y fecha especificada.

A continuación vamos a especificar el significado de los parámetros más importantes. El parámetro "host" contiene el nombre o la dirección IP[23] del servidor hacia donde se va a establecer la conexión con la base de datos. Si la base de datos se encuentra hospedada en la máquina local, el valor del argumento será "localhost". El argumento "dbname" indica el nombre de la base de datos dada por el recurso conexión de PostgreSQL. Los parámetros "user" y "password" especifican el nombre de usuario y la contraseña para acceder a la base de datos especificada en el argumento "dbname" [12].

Las conexiones entre el punto de origen y los puntos de destino (que pueden ser más de uno) están establecidas dentro de un archivo de configuración. Este archivo posee, en la primera línea, los parámetros de acceso a la base de datos desde la cual van a sincronizarse los registros, y en las líneas siguientes, la configuración de los parámetros de acceso a las bases de datos hacia las cuales van a llegar los registros sincronizados desde la base de origen. El programa que ejecuta el servicio de sincronización accede a este archivo y establece las conexiones respectivas desde la base de origen a las bases de destino.

Una vez establecidas las conexiones a cada punto del sistema distribuido se procederá a ejecutar el procedimiento que transfiere los datos por sincronizar de la base de origen a las bases de destino. Se ha preferido establecer un procedimiento general para sincronizar los datos para permitir la escalabi-

[23] La dirección IP (Internet Protocol) es una dirección única de 32 bits utilizada para identificar un nodo dentro de una red.

lidad del programa cuando se requiera añadir nuevas tablas que estén habilitadas para sincronizar. A continuación se describe el procedimiento que sincroniza los datos:

1. Se realiza una consulta a la tabla 'sys_tablas' para conocer cuáles son las tablas cuyos registros pueden sincronizar sus datos. Se ordenan según el campo 'nivel_prioridad'.

2. Por cada registro de tabla encontrado se asignan en variables temporales los valores obtenidos. Estos valores son:
   - Nombre de la tabla.
   - Campos de la tabla.
   - Campo(s) comparador(es).
   - Campo tipo de operación.
   - Campo fecha de operación.
   - Campo fecha de eliminación.
   - Campo tipo de prioridad.

3. Por cada registro de tabla encontrado, se consulta a la base de origen los registros que tiene su campo de operación marcado con "I" o "U" para ser sincronizados:
   - SELECT * FROM nombre_tabla WHERE tipo_operacion = 'I' OR tipo_operacion = 'U'

4. Si encuentra registros a ser sincronizados, se establece una consulta por cada uno de estos registros a la base de destino para verificar si existe el mismo registro en dicha base. Para realizar esta consulta se establece una restricción dinámica utilizando el(los) campo(s) comparador(es) de cada tabla. Por cada campo comparador, la restricción dinámica es:

   • restricción_dinámica -> campo_comparador = valor_origen [campo_comparador]

   La consulta en la base destino queda establecida de la siguiente forma:

   • SELECT * FROM nombre_tabla WHERE restricción_dinámica.

5. Si existe el registro en la base de destino, verifica si la operación a efectuarse es de inserción o actualización.

   • Si la operación es inserción, quiere decir que el registro no se puede sincronizar, debido a que éste ya existe en la base de destino, por lo que se ingresa el registro en el log de errores del sincronizador.

   • Si la operación es de actualización, se hace una verificación de si el mismo registro en la base destino ha sufrido alguna transacción desde la última vez que se sincronizó, lo cual se puede verificar a través del campo tipo de operación.

     ⇒ Si ese es el caso, se verifica el tipo de prioridad (campo tipo de prioridad) que posee la tabla a la que pertenece el registro, la cual puede ser para la primera transacción (se da prioridad al registro

modificado con mayor antigüedad) o para la última transacción (se da prioridad al registro más recientemente modificado). A partir de esto, se comparan los campos 'Fecha de operación' del registro de la base de origen con el mismo registro de la base de destino:

- ✓ Si la prioridad le pertenece al registro de la base de origen, entonces se almacena el registro de la base de destino en el log de errores de sincronización en la base destino, y se procede a la actualización del registro en la base destino.

- ✓ Si la prioridad le pertenece al registro en la base de destino, entonces no se podrá actualizar el registro en la base destino y se almacenará esta transacción fallida en el log de errores de sincronización.

⇒ En el caso de que el registro que se quiere actualizar en la base de destino no haya sufrido modificaciones desde la última vez que se sincronizó, se procede a la ejecución de la transacción en la base destino. En caso de que por algún tipo de falla no se complete la transacción, el registro es ingresado en el log de errores de sincronización.

6. En el caso de que no exista el registro en la base destino, el sincronizador intentará ingresar el registro:
   - INSERT INTO nombre_tabla VALUES (valor_baseorigen[campos_tabla])

   En caso de que no se pueda completar la transacción por algún tipo de falla, el registro es ingresado en el log de errores de sincronización.

7. Luego de finalizado el proceso en las bases destino, se hace una actualización del campo 'Operación' en los registros que fueron sincronizados o que no pudieron sincronizarse desde la base de origen y se desmarca dicho campo, lo cual indica que el servicio de sincronización ya atendió todos los requerimientos de cada tabla, sincronizando los que se pudieron y llevando un log de errores de los que no se pudieron. Para esto se vuelve a hacer una consulta a la tabla 'sys_tablas' de las tablas cuyos registros pueden sincronizar sus datos y se ejecuta la acción de actualización del campo 'Operación':
   - UPDATE nombre_tabla SET tipo_operacion = null WHERE tipo_operacion = 'I' OR tipo_operacion = 'U'

Para efecto de retroalimentación se ha generado una salida por pantalla de los resultados de la sincronización, incluyendo el log de errores. Adicionalmente, el log de errores se almacenará en un archivo de texto para decisiones posteriores al proceso de sincronización.

Finalmente, el sistema administrador de contenidos presenta una opción para que un administrador pueda configurar las tablas que pueden sincronizar.

Para esto, el usuario deberá ingresar:

- El nombre de la tabla.
- Los campos de la tabla, separados por punto y coma (;).
- El(los) campo(s) comparador(es). Si es más de uno, se los debe separar por punto y coma (;).
- El campo que indica el tipo de operación.
- El campo que indica la fecha de operación.
- El campo que indica la fecha de eliminación.
- El tipo de prioridad, si es para la primera o para la última transacción.
- El nivel de prioridad de sincronización de la tabla.

3.4. Diseño de interface con el usuario

En esta sección se definirán los flujos de ventanas con las opciones disponibles para los usuarios del sistema dependiendo del tipo de usuario al que pertenezcan, los niveles del administrador de contenidos y el diseño de la interacción con el usuario.

3.4.1. Flujo de ventanas y layouts

El flujo de ventanas describe la secuencia de las ventanas que el usuario observará a la vez que indican la funcionalidad requerida para el sistema. Permite ayudar a establecer la interfaz de usuario y la describe en términos de las tareas que se esperan que sean realizadas por éstos, sin necesidad de detallar todos los casos excepcionales, como el manejo de errores.

La Figura 3.7 describe el flujo de ventanas para un agente de bienes raíces, y la Figura 3.8 detalla el flujo de ventanas para un administrador del sistema.

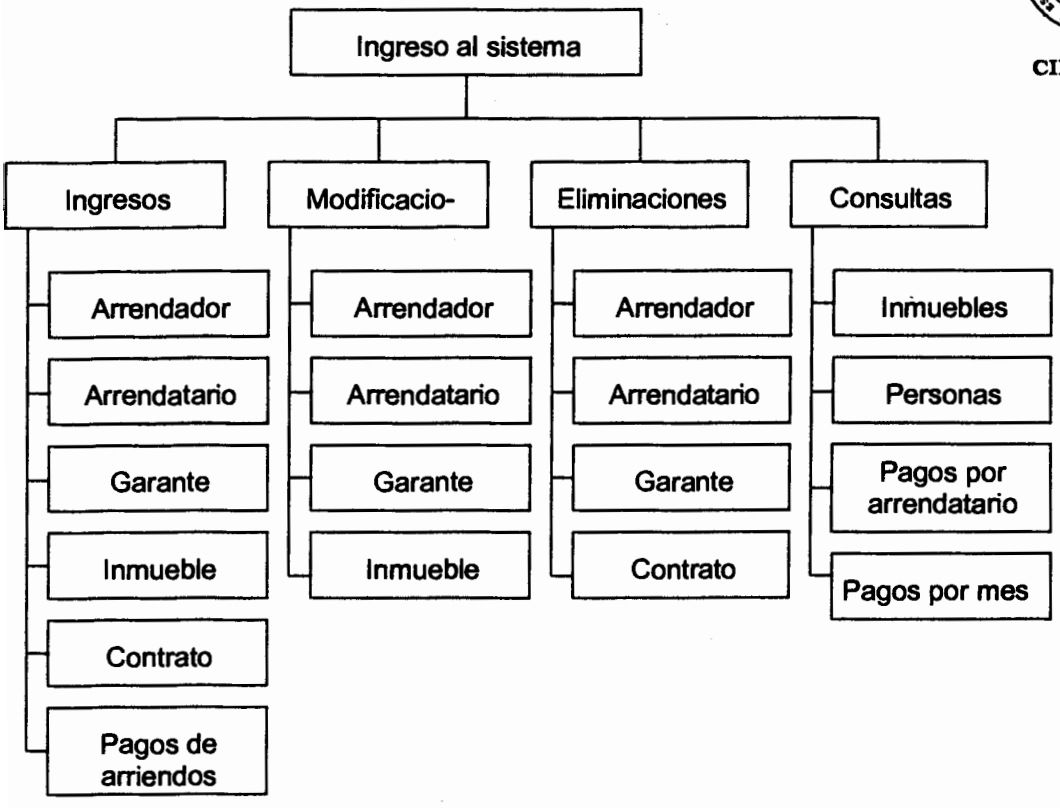


Figura 3.7. Flujo de ventanas para un agente de bienes raíces

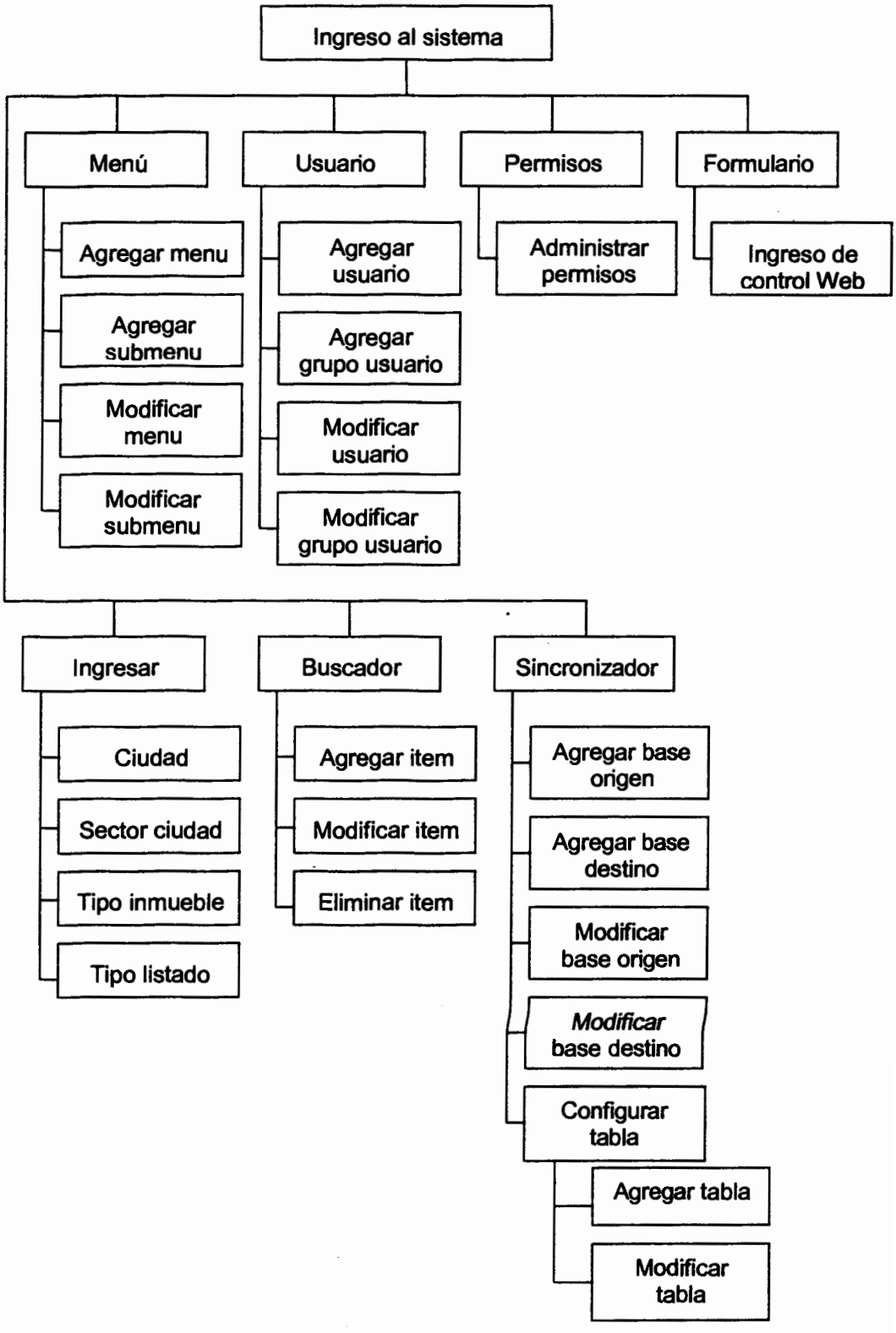


Figura 3.8. Flujo de ventanas para un administrador del sistema

Los layouts de cada ventana serán presentados dentro del apéndice A, que se refiere al manual de usuario.

El layout define la apariencia real de una ventana, describiendo sus partes visuales reales, como vistas, menús, botones, textos, campos, íconos, etc., y su ubicación dentro de una pantalla. Los layouts definen la conexión y secuencia de las ventanas.

3.4.2. Niveles del administrador de contenidos

El administrador de contenidos del sitio Web de este proyecto toma en cuenta 2 niveles para el manejo de la información del sitio: administración propiamente dicha, y la publicación de los contenidos.

La administración de contenidos está definida por el rol de los usuarios que definen qué información es la que se va a publicar y en qué lugar. Esta función es realizada por los administradores del sitio, quienes tienen las tareas de definir los roles de los usuarios del sistema, crear nuevas categorías y menús, crear nuevos tipos de contenidos (en este caso de los formularios para manejo de la información), configurar los criterios de seguridad y permisos de los grupos de usuarios, añadir y definir usuarios y tipos de usuarios [4].

La publicación de contenidos estará a cargo de los agentes de bienes raíces. Esta función les permite elaborar y publicar el contenido del sitio, agregando, modificando o eliminando registros que luego pueden ser consultados por los mismos agentes o por los usuarios públicos que navegan por el sitio, sin necesidad de tener conocimientos de HTML o cualquier otra herramienta técnica [4].

3.4.3. Diseño de la interacción con el usuario

A continuación presentamos el diseño de las páginas del sitio Web desarrollado para la demostración de este proyecto de tesis, el cual sigue las indicaciones previstas en el análisis de la interacción.

En primer lugar, todas las páginas deberán guardar consistencia en cuanto a la disposición de los elementos de la pantalla. Así, en la parte superior estará fijo el logo del sitio, mientras que en el lado izquierdo se ubicará el menú de opciones. El contenido de este menú dependerá del usuario que esté navegando el sitio, y en la página de inicio aparecerá el menú de los usuarios públicos, es decir aquellos que no han iniciado sesión.

Adicionalmente, en esta página se ubicará la opción de inicio de sesión debajo del menú de opciones, donde se pide ingresar el nombre de usuario y la contraseña. En el lado derecho se visualizará un buscador de inmuebles, en

La publicación de contenidos estará a cargo de los agentes de bienes raíces. Esta función les permite elaborar y publicar el contenido del sitio, agregando, modificando o eliminando registros que luego pueden ser consultados por los mismos agentes o por los usuarios públicos que navegan por el sitio, sin necesidad de tener conocimientos de HTML o cualquier otra herramienta técnica [4].

3.4.3. Diseño de la interacción con el usuario

A continuación presentamos el diseño de las páginas del sitio Web desarrollado para la demostración de este proyecto de tesis, el cual sigue las indicaciones previstas en el análisis de la interacción.

En primer lugar, todas las páginas deberán guardar consistencia en cuanto a la disposición de los elementos de la pantalla. Así, en la parte superior estará fijo el logo del sitio, mientras que en el lado izquierdo se ubicará el menú de opciones. El contenido de este menú dependerá del usuario que esté navegando el sitio, y en la página de inicio aparecerá el menú de los usuarios públicos, es decir aquellos que no han iniciado sesión.

Adicionalmente, en esta página se ubicará la opción de inicio de sesión debajo del menú de opciones, donde se pide ingresar el nombre de usuario y la contraseña. En el lado derecho se visualizará un buscador de inmuebles, en

donde los criterios de búsqueda son establecidos por el administrador de contenidos del sistema. En el sector central estará la información de interés; en el caso de la página de inicio, se mostrarán las ofertas más recientes de inmuebles que se encuentran disponibles para ser alquilados y a los cuales se puede acceder por mayor información.

En la Figura 3.9 se muestra la división de los sectores de las pantallas del sitio Web, tal como ha sido descrito en los párrafos anteriores.

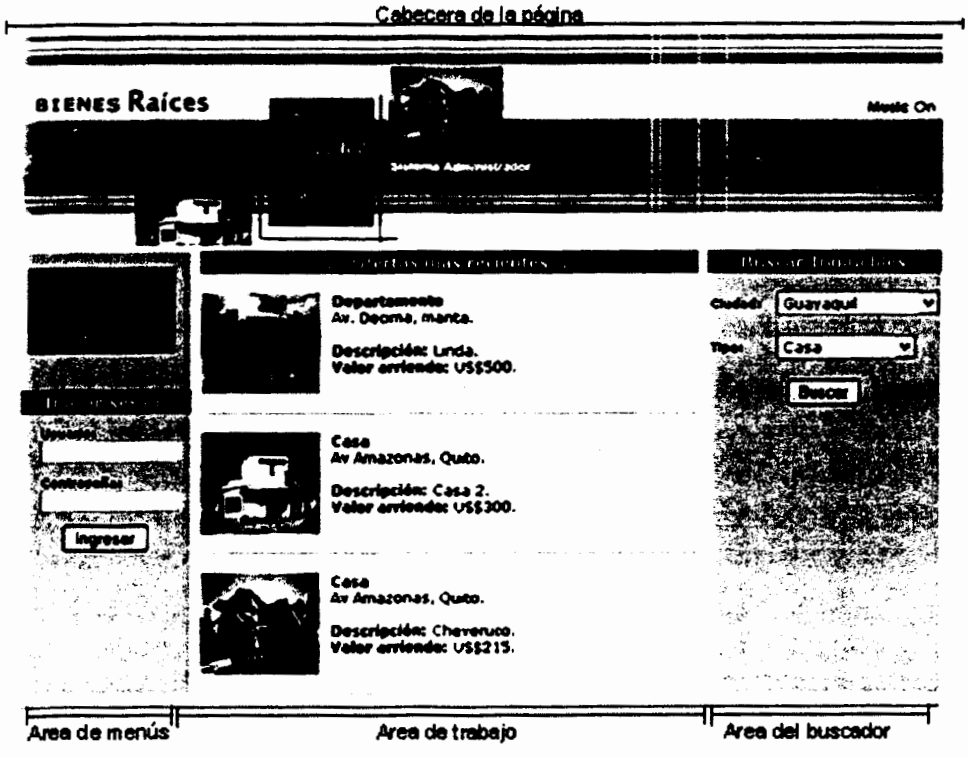


Figura 3.9. Areas en las que se dividen las pantallas del sitio

Como se puede apreciar en la Figura 3.9, en el área de trabajo aparecen las ofertas más recientes, las cuales incluyen una imagen del inmueble acompañada de una breve descripción y el valor de su arriendo, que son las características que les interesan a las personas que navegan por el sitio.

Asimismo, para que los usuarios desarrollen un buen modelo mental se ha hecho uso de colores que contrasten para poder diferenciar las áreas en las que se dividen las pantallas del sitio, así como colores diferenciados para el menú, y los títulos de cada área, todo esto con el fin de asegurar la visibilidad del sistema.

Es importante anotar que los usuarios para observar los submenús deben pasar el cursor del mouse por encima de las opciones del menú para que aparezcan los primeros, tal como se aprecia en la Figura 3.10.

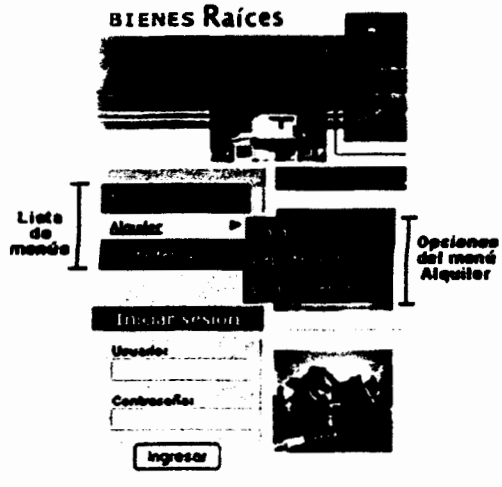


Figura 3.10. Menús y submenús del sistema

Con la presencia de los menús en todas las páginas se asegura la navegabilidad del sitio, ya que permiten pasar de una opción a otra en un solo paso. Los nombres de las opciones deben cumplir con el principio de permisividad explicado en el análisis de la interacción Hombre – Máquina.

Cuando un usuario registrado en el sistema haya ingresado se le presentará la pantalla que se muestra en la Figura 3.11. En esta página el título del área de trabajo indica que el usuario se encuentra en el módulo de administración del sistema y se muestran el nombre del usuario y su rol. En el área de menús aparecen las opciones a las que el usuario tiene acceso dependiendo del grupo al que pertenece, y se presenta el enlace para cerrar la sesión.

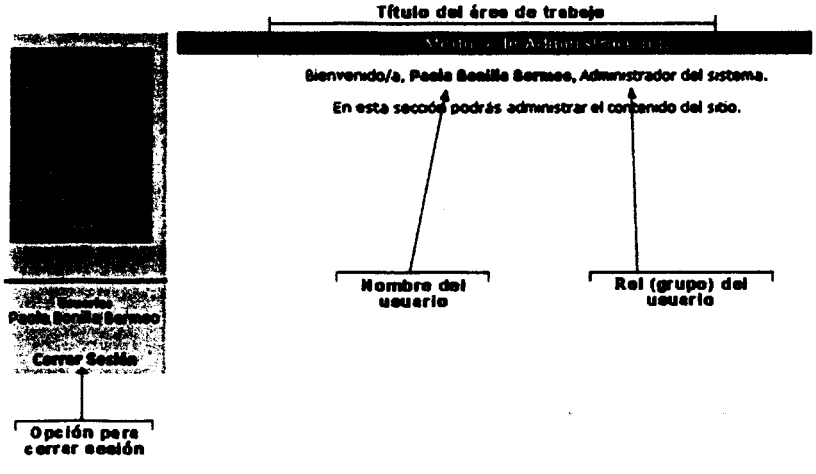


Figura 3.11. Pantalla de bienvenida al usuario que ingresa al sistema

Cuando el usuario hace uso de una de las opciones del menú, se le presentará un formulario que le permitirá interactuar con el sistema. Los nombres de los campos de los formularios proporcionarán una terminología que cum-

plan con el principio de la familiaridad, para que los usuarios reconozcan fácilmente el uso de cada uno de estos campos.

Los formularios serán ubicados en el área de trabajo, tal como se lo puede apreciar en la Figura 3.12. En la parte superior, el título deberá indicar la transacción que se va a realizar. Luego se mostrarán todos los campos que pertenecen al formulario, y finalmente el botón que permitirá ejecutar la acción requerida. Este tipo de diseño será utilizado para todos los formularios del sistema.

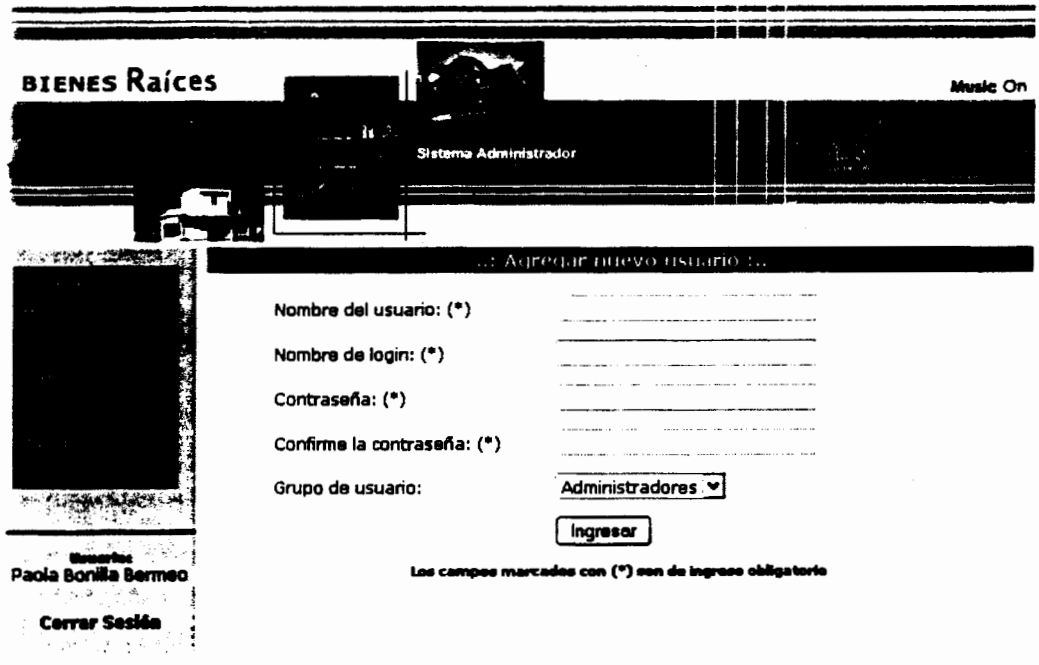


Figura 3.12. Pantalla de formulario

Cuando la transacción ha sido realizada, el resultado de la misma aparecerá con un mensaje de realización o error. Este mensaje deberá ser lo suficien-

temente entendible para permitirle al usuario obtener una buena retroalimentación y reconocimiento de errores, dado el caso, de la acción ejecutada. Un ejemplo de esto se lo puede observar en las Figuras 3.13 y 3.14.

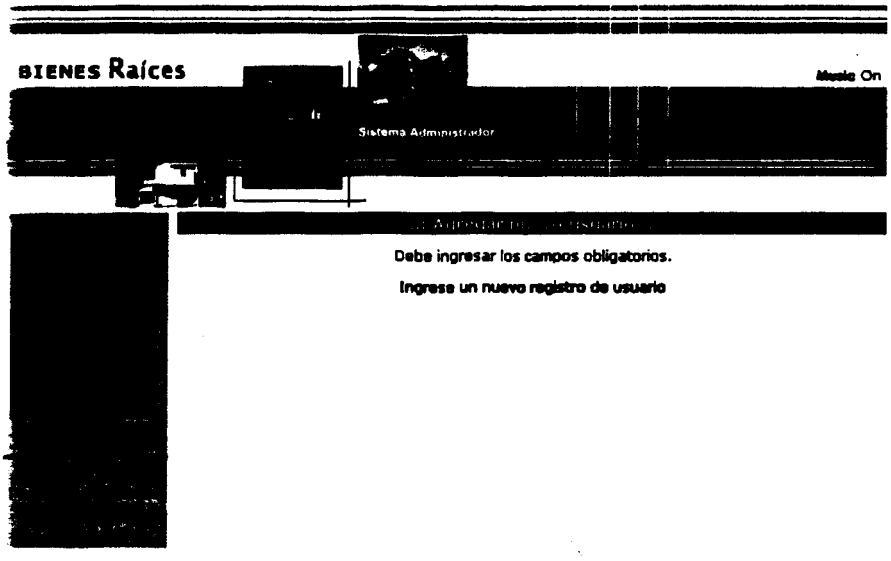


Figura 3.13. Pantalla con mensaje de error

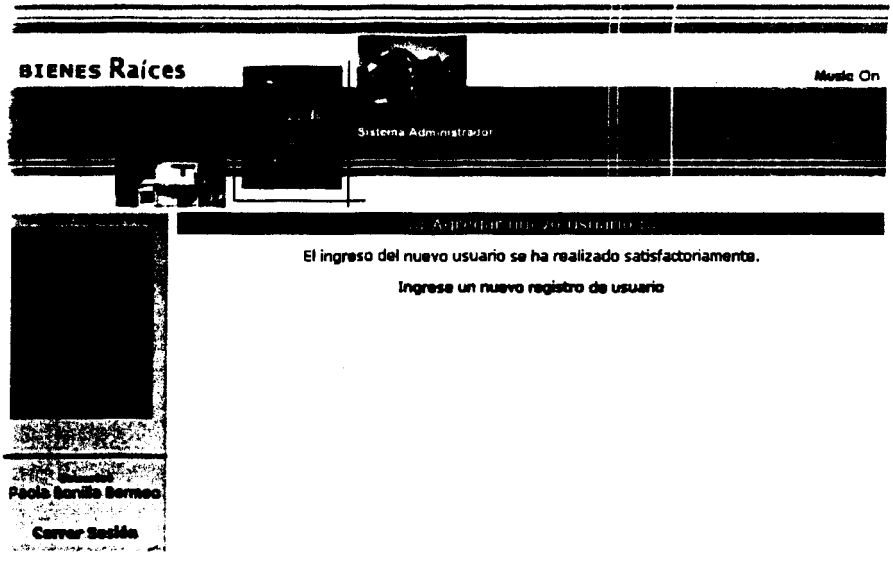


Figura 3.14. Pantalla que muestra una transacción satisfactoria

Por último, en caso de que un usuario haya realizado una operación de consulta, los resultados serán presentados indicando la cantidad de registros encontrados y las características de la consulta, tal como lo muestra la Figura 3.15.

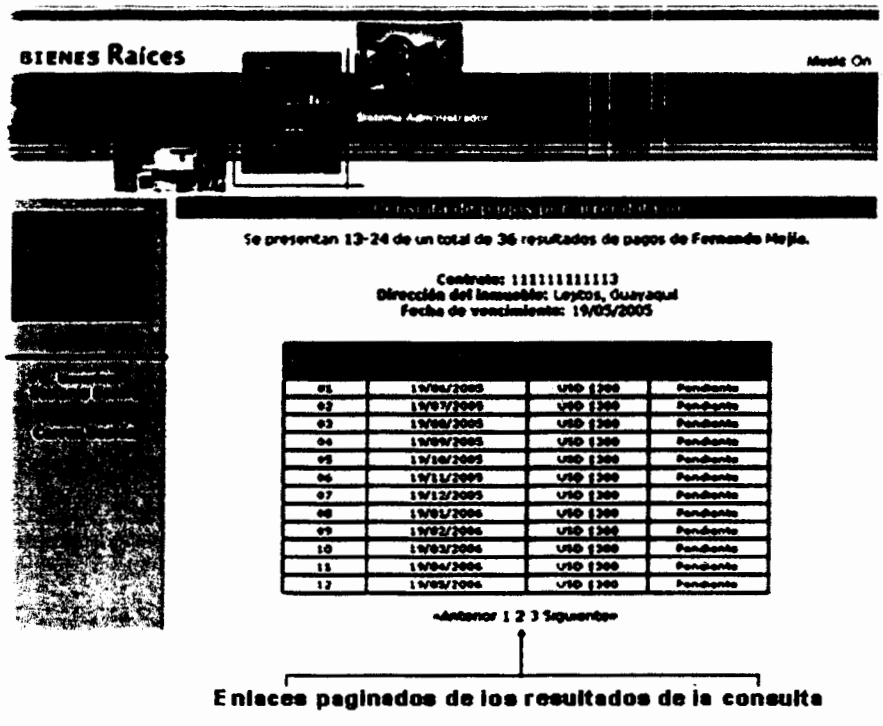


Figura 3.15. Pantalla con resultado de consulta

De la misma forma, como se aprecia en la Figura 3.15, si la consulta proporciona más de 8 resultados, éstos serán presentados en varias páginas, y en la parte inferior del área de trabajo serán mostrados los enlaces a cada una de las páginas que contienen los resultados de la consulta.

CAPITULO

Implementación y pruebas

CAPÍTULO 4

IMPLEMENTACIÓN Y PRUEBAS

4.1. Procesos de implementación

En esta sección se hace una descripción del proceso de implementación de la aplicación demostrativa de este proyecto de tesis, una vez realizadas las etapas de análisis y diseño del sistema.

Se han definido ciertas reglas en la programación de la aplicación para facilitar el mantenimiento y entendimiento del sistema. Las reglas adoptadas son:

- Los nombres de las clases empiezan con letra mayúscula y deben ser significativos.
- Los nombres de los métodos empiezan con letra mayúscula por cada palabra que la compone, por ejemplo, CrearContrato.
- Los nombres de las variables de clase y método empiezan con letra minúscula.
- El código debe estar debidamente documentado, incluyendo descripciones para las clases y los métodos.

Tal como se explicó en el análisis de tecnologías, el lenguaje de programación escogido para el desarrollo del proyecto es PHP, que es un lenguaje especialmente enfocado en el desarrollo de aplicaciones Web con contenido dinámico.

El diseño de las páginas Web, tal como se definió en la sección de herramientas de desarrollo, fue hecho en Macromedia Dreamweaver, siguiendo los principios de diseño de la interfaz establecidos en el capítulo anterior.

Dentro del proceso de instalación y configuración de las herramientas de desarrollo, en primer lugar se procedió a la instalación del servidor Web Apache y del módulo de PHP. El siguiente paso dado fue la configuración de PHP para que funcione como un módulo en Apache. Adicionalmente se instaló el IDE OpenSource Zend Development Environment para el desarrollo de la lógica de programación, y la base de datos PostgreSQL para el almacenamiento de los datos.

Luego de realizado el proceso de instalación y configuración de las herramientas de desarrollo, se procedió a la codificación del módulo de conexión a la base de datos desde una página PHP. Las pruebas de conexión se realizaron utilizando el Zend Development Environment y programando la clase Database para que establezca la conexión a la base de datos.

A continuación se programaron las páginas y las clases en PHP, haciendo pruebas de las transacciones de inserción, modificación, eliminación y consulta de registros desde y hacia la base de datos. De igual forma, se desarrolló el servicio de sincronización de datos en PHP, estableciendo las conexiones a las bases de datos origen y destinos. La base de datos origen es la que envía los datos y actúa como publicador, mientras que una base de datos destino es la que recibe los datos y actúa como suscriptor del sistema distribuido.

Luego del desarrollo de la lógica de programación se realizaron las pruebas para comprobar el correcto funcionamiento del sistema y eliminar los errores que no habían sido detectados.

4.2. Plan de pruebas

En esta sección se hace una descripción general de las estrategias de pruebas realizadas al sistema para garantizar su correcto funcionamiento.

En primer lugar se tienen las pruebas de unidad, las cuales se centran en cada módulo implementado en el código y prueban los recorridos importantes de cada uno de ellos. Para las pruebas de unidad se toman en cuenta los siguientes aspectos:

- Se debe asegurar que el flujo de la información se produzca de manera adecuada hacia y desde el módulo que es objeto de la prueba a través de la interfaz del sistema.
- Las estructuras de datos del módulo deben conservar los datos que contienen durante toda la ejecución del recorrido.
- Se debe probar todos los caminos de las estructuras (bucles) de control para asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez.
- Probar los casos de error que pueden producirse en el recorrido a través del módulo.

Adicionalmente las pruebas se encaminan a encontrar errores en inicializaciones de datos erróneas, inconsistencias en los tipos de datos, nombres de variables erróneos, comparaciones incorrectas y finalización inesperada de bucles.

Una vez realizadas las pruebas de unidad se efectúan las pruebas de integración, las cuales permiten integrar los módulos probados en unidad y probar todo el sistema en conjunto con el objetivo de detectar errores asociados con la interacción entre los módulos del programa.

En este plan de pruebas se ha planeado utilizar la técnica de la integración descendente, en donde se va probando desde el módulo de ingreso al sis-

tema, donde se crea la sesión de usuario, y luego se sigue la secuencia incorporando los módulos de ingresos, modificaciones, eliminaciones, pagos, consultas, permisos y manejo de formularios.

Después de que las pruebas de integración estuviesen finalizadas se continúa con las pruebas de validación, en las cuales se comprueba que el sistema cumple con los requerimientos funcionales, que se logran los requisitos de seguridad, confiabilidad, facilidad de mantenimiento y que la documentación del código sea correcta y entendible. Es importante destacar que este tipo de pruebas habitualmente deben ser realizadas por el usuario final, pero en este caso se ha obviado la existencia de los mismos, por lo que las pruebas de validación han sido realizadas por nosotros mismos.

En los casos en que se encontraran errores en el sistema como producto de la ejecución de un recorrido dentro del plan de pruebas, se debe llevar a cabo un proceso de depuración. La depuración del código comienza luego de que un caso de prueba encuentra un error, se evalúan los resultados obtenidos con los esperados, se corrige el error y se vuelve a probar.

4.3. Resultados de las pruebas

Dentro de las pruebas de unidad se encontraron errores relacionados con inicializaciones de datos erróneas o inexistentes, nombres de variables mal escritos o inexistentes, comparaciones incorrectas y finalización inesperada de bucles.

Con respecto a las pruebas de integración, se hizo un chequeo exhaustivo del sistema haciendo un recorrido de cada uno de los escenarios posibles. Estas pruebas finalizaron una vez que ya no se detectaron errores en el sistema ya integrado.

Finalmente, se comprobó que el sistema cumpla con los requerimientos bajo los cuales fue desarrollado, cubriendo de esta manera las pruebas de validación del sistema.

CONCLUSIONES Y RECOMENDACIONES

- Como resultado de haber desarrollado este proyecto de tesis tenemos un sistema administrador de bienes raíces que, principalmente, combina una aplicación para administrar los contenidos de su sitio Web, con un servicio que sincroniza todas las transacciones generadas en cualquiera de los servidores que forman parte del negocio. Para esta tesis, hemos enfocado el sistema hacia negocios en los cuales se maneje flujo de información muy variable y en donde se requiera optimizar recursos como tiempo y costos.

- Con el fin de cumplir con el objetivo de asegurar la independencia de plataforma que debe poseer este proyecto, la solución se implementó en herramientas que aseguraron la portabilidad y escalabilidad del mismo.

- El código de la aplicación ha sido diseñado y estructurado de tal manera que facilita la implementación de nuevos módulos al sistema de forma rápida y sencilla.

- La interfaz del sistema ha sido diseñada con el fin de brindar al usuario facilidad en la navegación del sitio y en el entendimiento de las tareas que puede realizar.

- La base de datos fue desarrollada de tal forma que se logró mantener la consistencia de datos y la integridad relacional de las tablas partícipes de la sincronización de los registros.

- El dividir el proyecto en dos modelos de datos, uno de negocios y otro de contenidos, permite separar los aspectos relacionados a cualquier negocio en sí, en este caso el de bienes raíces, de todo lo relacionado con la presentación de contenidos que pueden ser aplicados a cualquier tipo de negocios, y de esta forma se combinan dos poderosas herramientas en una misma aplicación.

- La solución encontrada para la resolución de conflictos de clave primaria, utilizando estratégicamente claves compuestas, en conjunto con otros campos específicos de tabla, como la fecha de operación, el campo comparador y el tipo de operación, mantuvo la integridad de los datos y eliminó en su mayoría los posibles problemas al momento de sincronizar la información. Cabe recalcar, que todo fallo producido durante el proceso de sincronización, se reporta al registro de errores para decisiones manuales posteriores. De esta manera, queda registrada toda transacción del sistema sin que haya pérdida de datos..

- El tratamiento de imágenes utilizando una tabla específica para este fin, que divide la imagen en fragmentos de 1024 bytes, aseguró la re-

ducción de los tiempos de consulta y facilitó la transmisión de datos al momento de sincronizar los mismos.

- Los niveles establecidos en la administración de contenidos, aseguraron una división apropiada de las tareas para cada usuario y a la vez proporcionaron la seguridad necesaria para evitar la manipulación de información restringida por parte de usuarios no autorizados.

- El crear un archivo para la configuración de las bases de datos dentro del modelo distribuido, que no forme parte de la base local, ni de la implementación del sitio Web, permitió la independencia de las operaciones del servicio de sincronización y, adicionalmente, brindó flexibilidad para la creación de nuevos puntos de destino.

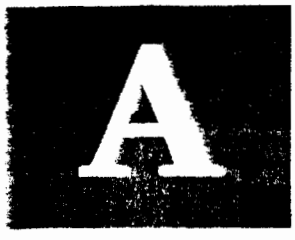
- La implementación del servicio de sincronización bajo la misma plataforma en la cual fue desarrollado el sitio Web, permitió el uso del conocimiento adquirido en la realización de este proyecto y adicionalmente, aseguró la independencia de plataformas. Por otra parte, con el uso de una tabla del sistema, el sincronizador brinda la flexibilidad a un usuario administrador de configurar este servicio de acuerdo a las necesidades del negocio.

- El sistema ha sido desarrollado de tal forma que le permite a los nuevos usuarios la posibilidad de extenderlo conforme exija el negocio. Por lo tanto, es recomendable el desarrollo de proyectos futuros que tomen como referencia esta tesis.

- Es recomendable generar un módulo que recorra todos los registros que se encuentren inactivos e inutilizados durante un determinado período, para eliminarlos físicamente de la base de datos, puesto que en esta tesis se utilizó solo el concepto de eliminación lógica de registros.

- Para versiones futuras, se recomienda el uso de procedimientos estándares de seguridad en la transferencia de los registros que están siendo sincronizados, con el fin de que no se transmitan como texto plano, debido a que esto podría ser vulnerable a ataques externos.

- Adicionalmente, se sugiere que la información transmitida pueda ser enviada en un solo paquete en cada proceso de sincronización, en vez de esperar una respuesta por cada registro sincronizado.



Apéndices

APÉNDICE A

MANUAL DE USUARIO

En este apéndice se detallarán las guías para el uso del Sistema Administrador de Bienes Raíces. Para poder ingresar a la página principal del sitio se debe utilizar un navegador de Internet y acceder a la dirección:

http://<nombreDominioServidor>/BienesRaices

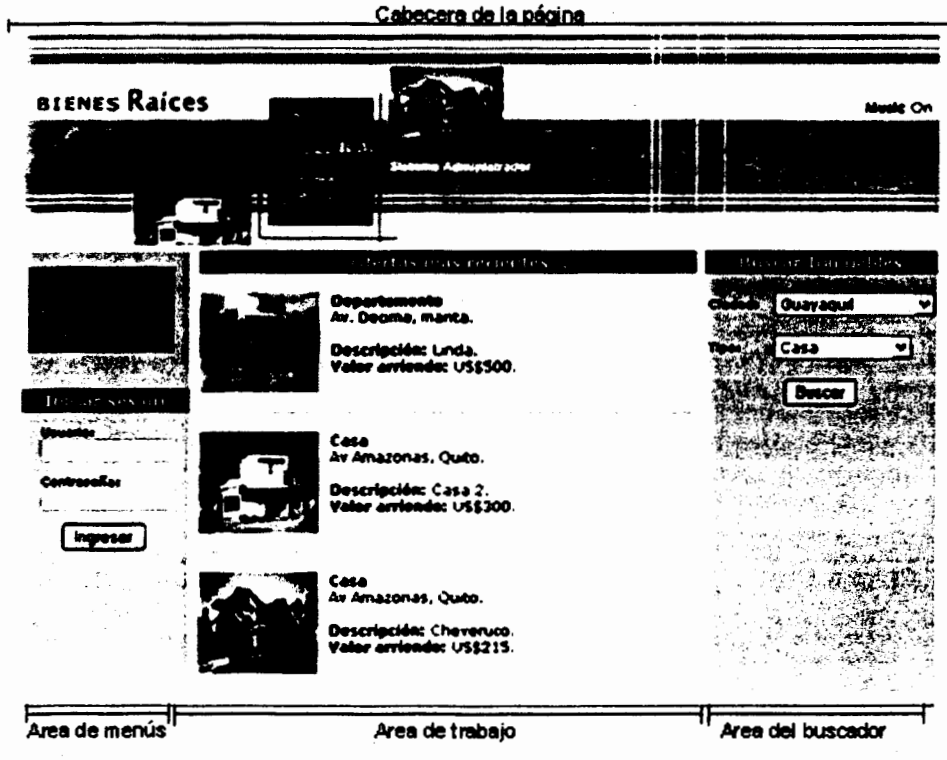


Figura A.1. Pantalla de inicio del sitio de Bienes Raíces

Esta pantalla presenta las ofertas más recientes de inmuebles disponibles, el menú de opciones para los usuarios que navegan por el sitio sin haber ingresado con un nombre de usuario y un buscador de inmuebles (Figura A.1). El menú para los navegadores del sitio que no hayan accedido al sistema tienen las opciones de "Inicio", "Alquiler" y "Contáctenos".

En el menú "Alquiler" se tienen las opciones de "Casa", "Departamento" y "Local comercial". Al escoger una de estas opciones, se obtiene un listado de los inmuebles que están disponibles según el tipo escogido, con una información resumida de cada uno de los inmuebles presentados (Figura A.2). Esta información resumida contiene la dirección del inmueble, una breve descripción y el valor por el alquiler del mismo.

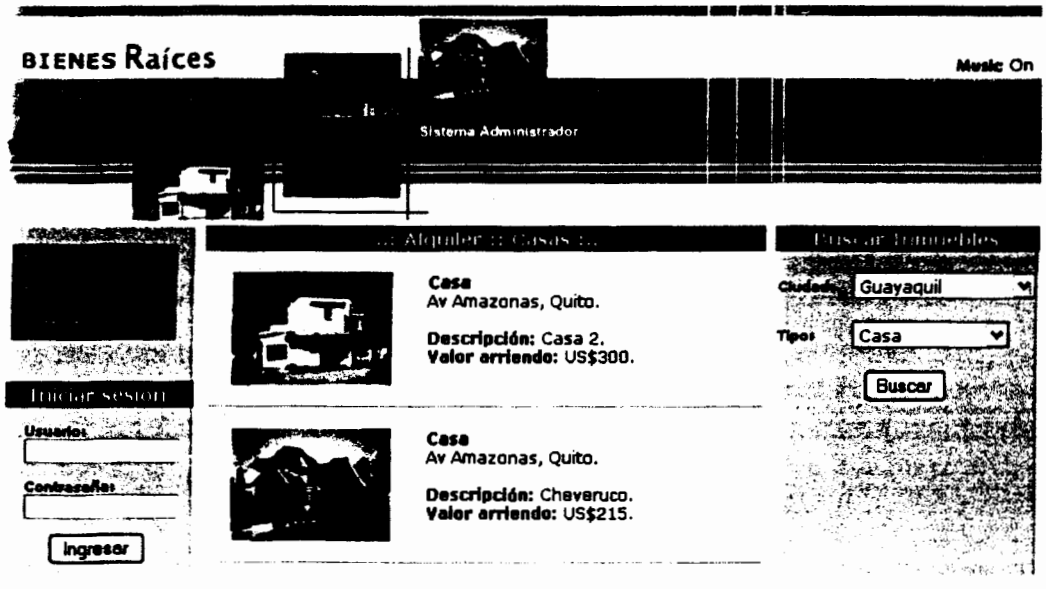


Figura A.2. Listado de inmuebles de un submenú

Información sobre los inmuebles

Para acceder a una información más completa de un inmueble específico, se debe presionar la imagen del inmueble o el título que indica el tipo del inmueble al inicio de la información resumida de cada inmueble (Figura A.3). Esta información incluye la ubicación del inmueble, una descripción breve, el monto del arriendo, sus dimensiones y el número de dormitorios y baños que posee.

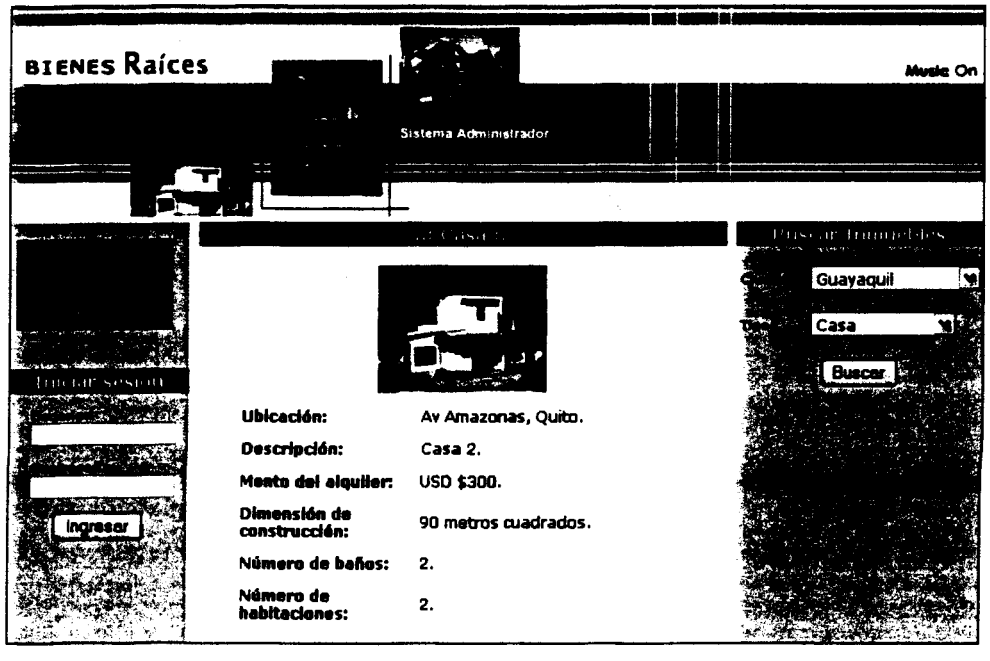


Figura A.3. Información detallada del inmueble seleccionado

Utilización del buscador

Se ha implementado un servicio que permite a los usuarios consultar por inmuebles en una ciudad determinada y restringiendo la búsqueda según el

tipo de inmuebles que se requiera. Los resultados de la búsqueda (Figura A.4) proporcionan información sobre el total de resultados encontrados y presentados en cada pantalla, y sobre la ubicación, descripción y valor del arriendo de cada uno de los inmuebles presentados, proporcionando enlaces para consultar por la información específica de cada uno de ellos.

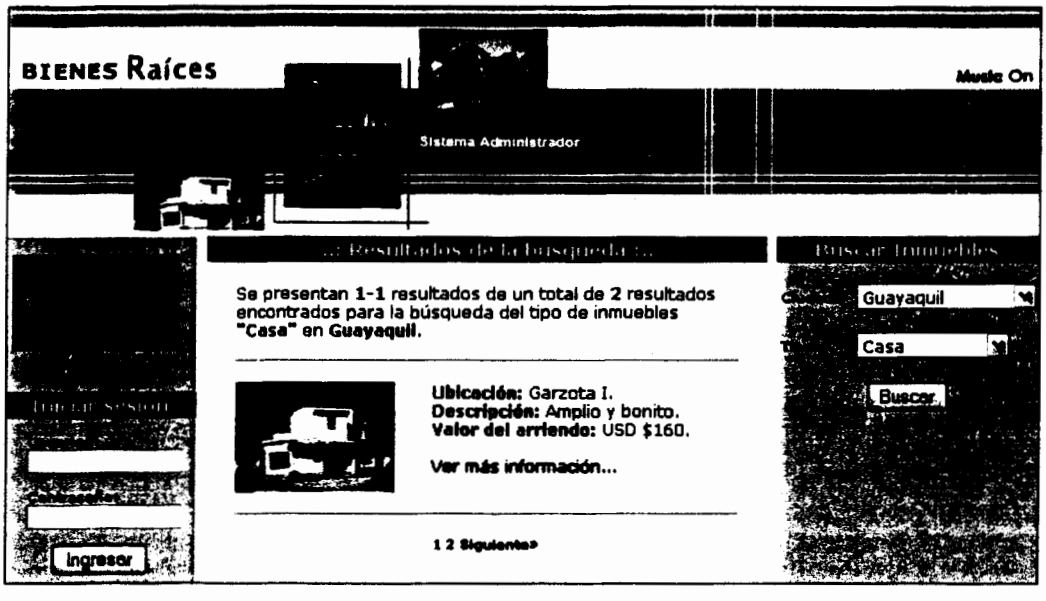


Figura A.4. Resultados de la búsqueda

Acceso autorizado al sistema

Los usuarios que tienen acceso autorizado a la administración del contenido del sitio deben ingresar su nombre de usuario y contraseña en la parte que indica el inicio de sesión de bajo del menú de opciones del sistema (Figura A.5).

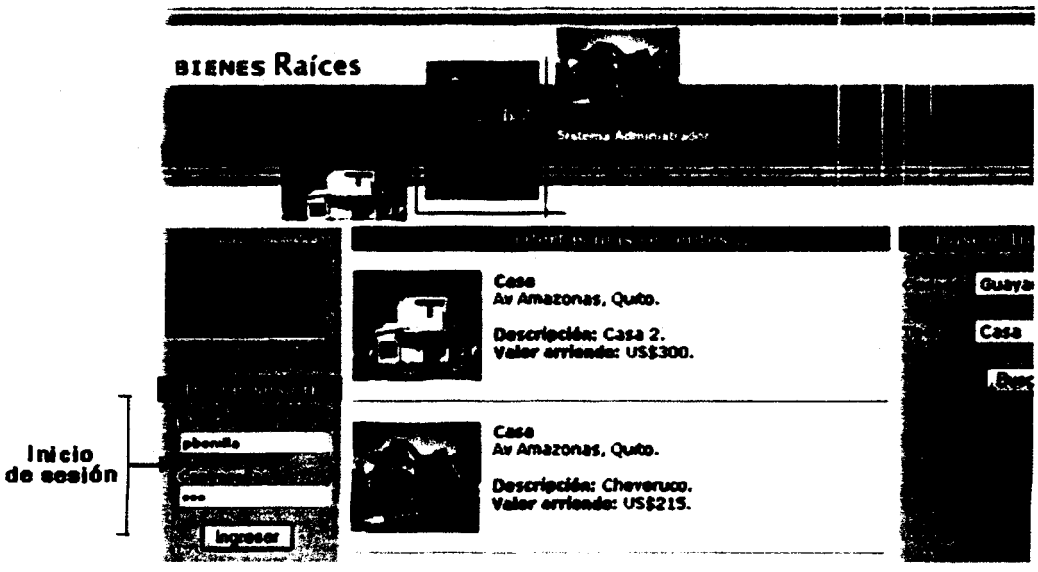


Figura A.5. Inicio de sesión

Si el nombre de usuario o la contraseña son incorrectos, se presentará un mensaje de error (Figura A.6), sino se dará paso a la página de bienvenida de los usuarios registrados (Figura A.7), la cual indica el nombre del usuario y describe al grupo al cual pertenece, además presenta el menú de opciones según su grupo de usuario.

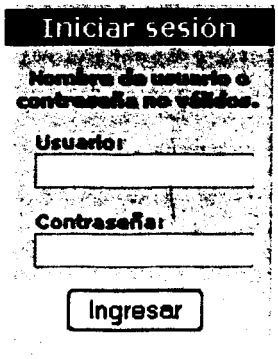


Figura A.6. Error en nombre de usuario o contraseña

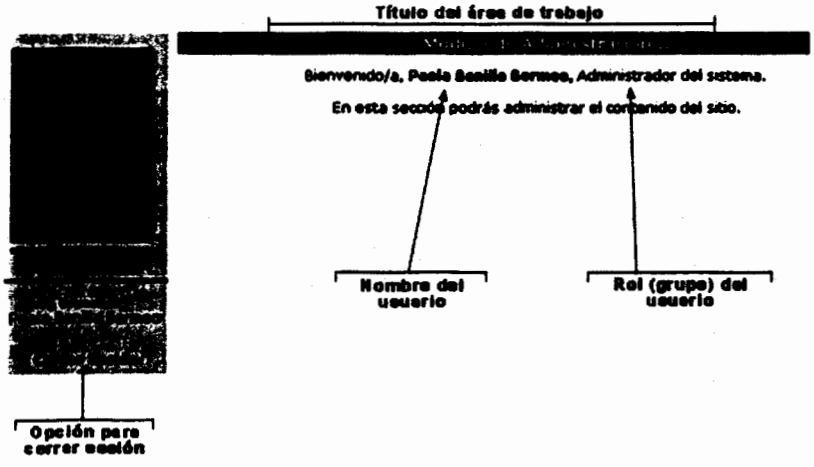


Figura A.7. Pantalla de bienvenida de usuario autorizado

Opciones de agentes de bienes raíces

Los usuarios del grupo de agentes de bienes raíces tienen las opciones de:

1. Ingresar.
2. Modificar.
3. Consultar.
4. Eliminar.
5. Pagos.

Ingreso de un registro.- *Un agente tiene permitido el ingreso de registros* de: arrendadores, arrendatarios, garantes, inmuebles, contratos de arrendamientos, ciudades, sectores de ciudades, y tipos de inmueble (Figura A.8). Para ingresar un nuevo registro, el usuario debe llenar todos los campos ne-

cesarios de los formularios de ingreso y presionar el botón Ingresar para efectuar la operación.

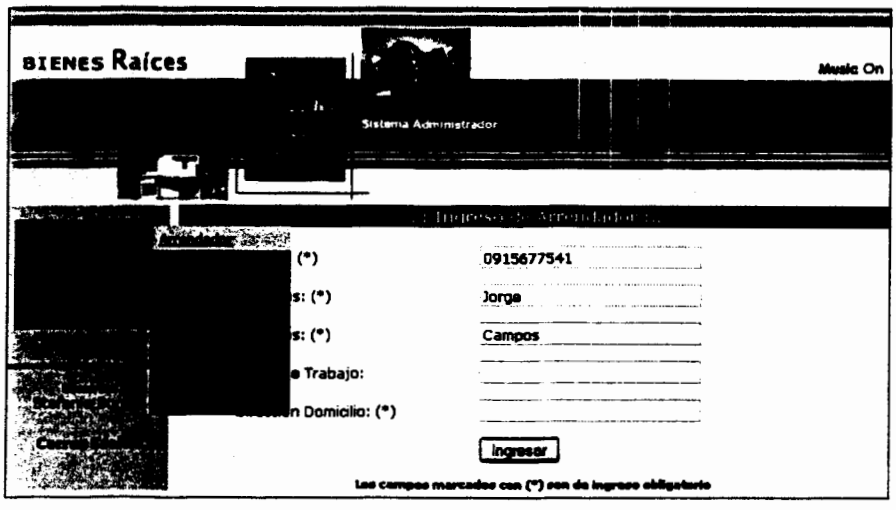


Figura A.8. Formulario de ingreso de registro

En caso de que ocurra algún error aparecerá un mensaje indicando el mismo (Figura A.9), y en caso de éxito, se indicará que la transacción se realizó satisfactoriamente (Figura A.10). Estas pantallas de mensajes son presentadas en todas las transacciones del sistema que requieren el ingreso, actualización o eliminación de datos.

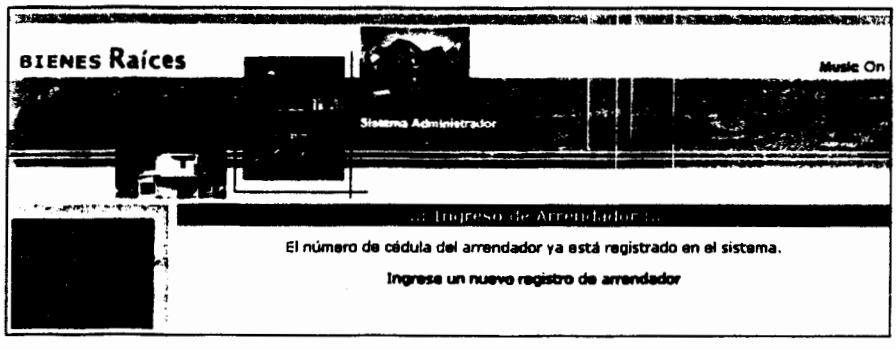


Figura A.9. Mensaje que indica un error en la transacción

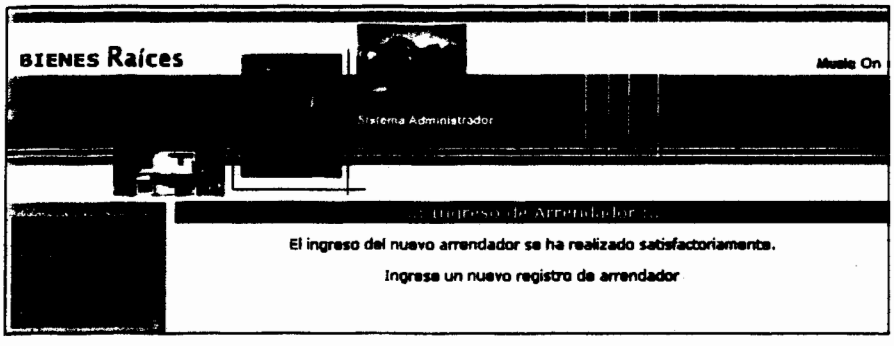


Figura A.10. Mensaje que indica el resultado de la transacción

Modificación de un registro.- Un agente tiene permitido la modificación de datos de registros de: arrendadores, arrendatarios, garantes, inmuebles, ciudades, sectores de ciudades y tipos de inmuebles. Primero se presenta una pantalla en donde se debe ingresar o seleccionar un campo clave (Figura A.11) y se debe presionar el botón Consultar. Luego aparece el formulario que muestra los datos del registro seleccionado, permitiendo al usuario cambiar la información presentada, tras lo cual debe presionar el botón Modificar para efectuar la operación, cuyo resultado aparece en un mensaje posterior.

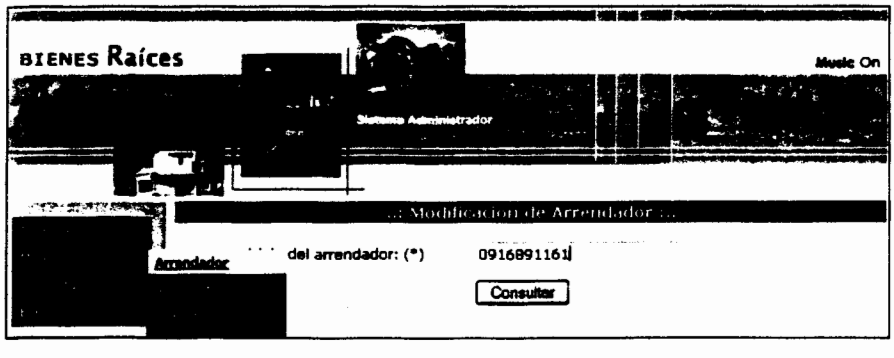


Figura A.11. Ingreso o selección de un registro para modificar sus datos

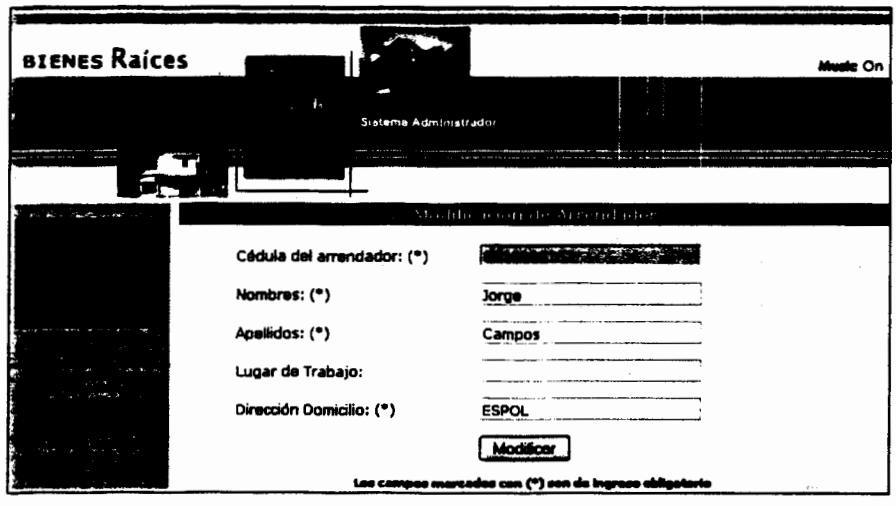


Figura A.12. Formulario de modificación de datos de un registro

Eliminación de un registro.- Un agente tiene permitido la eliminación de registros de: arrendadores, arrendatarios, garantes y contratos. Primero se presenta una pantalla en donde se debe ingresar o seleccionar un campo clave (Figura A.11) y se debe presionar el botón Consultar. Luego se muestran los datos del registro seleccionado (Figura A.13), y si el usuario lo desea eliminar debe presionar el botón Eliminar para efectuar la operación, cuyo resultado aparece en un mensaje posterior.

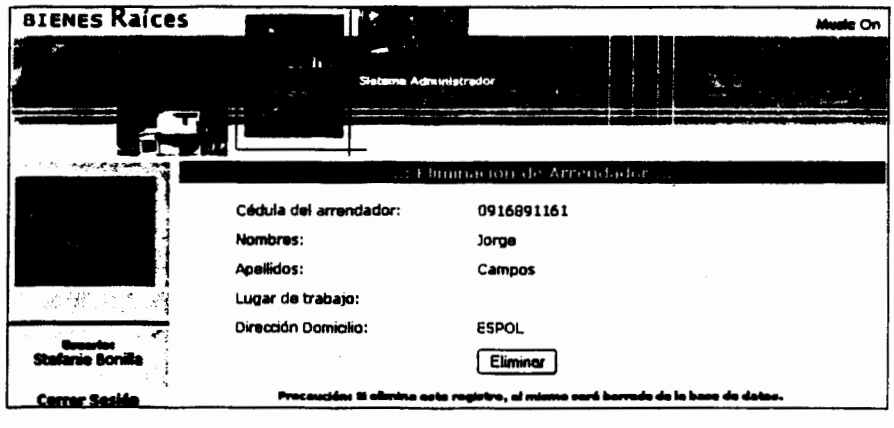


Figura A.13. Pantalla de eliminación de un registro

Consultas.- Un agente tiene permitido la consulta de: pagos de arriendos por mes, pagos de arriendos por arrendatarios, inmuebles y personas ingresadas en el sistema. Para consultar información del sistema el usuario debe llenar o seleccionar los campos necesarios en el formulario de consulta y presionar el botón Consultar para efectuar la operación (Figura A.14).

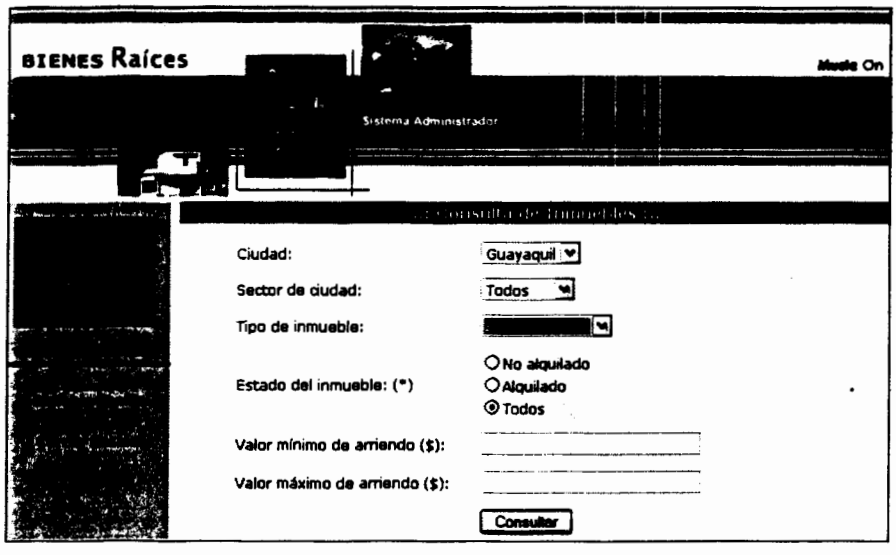


Figura A.14. Formulario de consulta de información

Luego se presentan los resultados de la consulta. En caso de no hallarse resultados se indica dicha situación (Figura A.15).

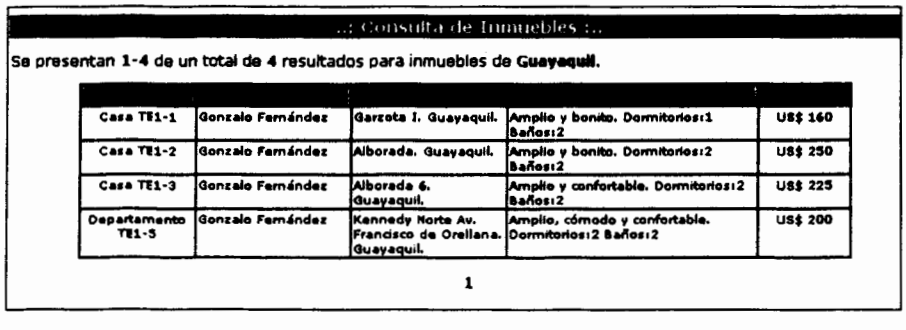


Figura A.15. Resultados de una consulta

Pagos de arriendos.- Un agente tiene permitido registrar el pago de un arriendo por parte de un arrendatario, para lo cual debe introducir el número del contrato, tras lo cual aparecerán los datos del mismo, indicando la fecha a la cual corresponde realizar el pago del alquiler (Figura A.16).

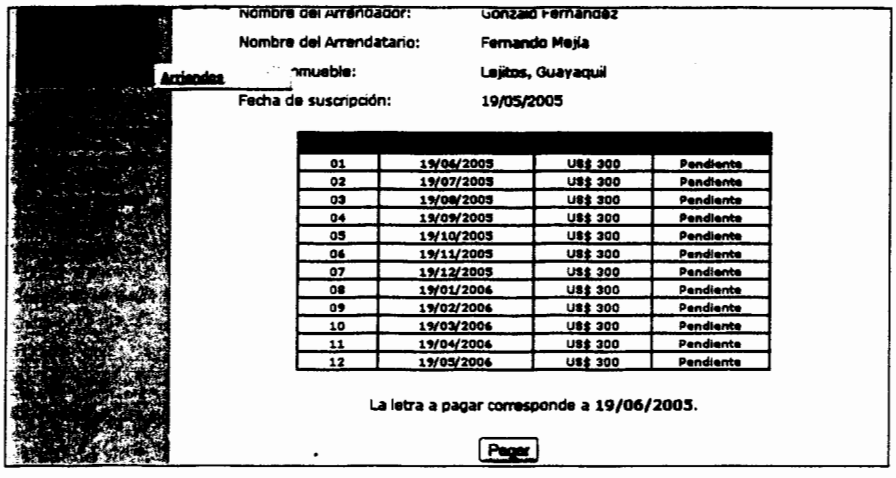


Figura A.16. Pantalla de pago de arriendo

APÉNDICE B

MANUAL DEL ADMINISTRADOR

En este apéndice se detallará la guía para la administración, instalación y configuración del sistema.

Opciones de administradores del sistema

Los usuarios del grupo de administradores tienen las opciones de:

1. Usuarios.
2. Menús.
3. Permisos.
4. Formularios.
5. Ingresar.
6. Buscador.
7. Sincronizador.

Usuarios.- Esta opción permite añadir nuevos usuarios y grupos de usuarios al sistema y modificar sus respectivos datos (Figura B.1). Estas opciones funcionan de la misma manera que los ingresos y modificación de registros explicados anteriormente (Apéndice A).

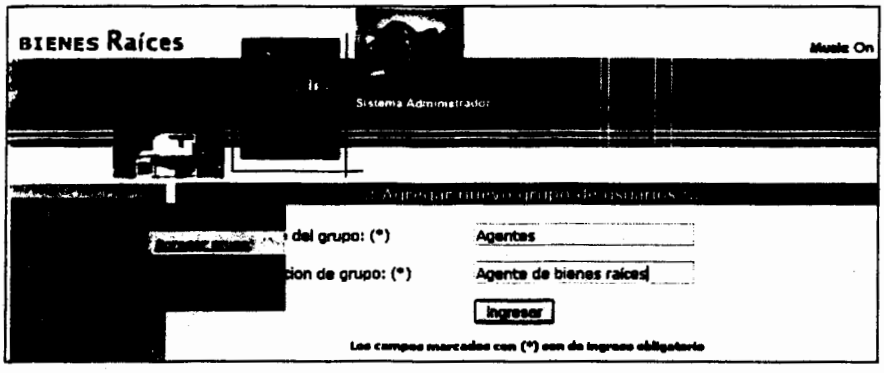


Figura B.1. Pantalla de administración de usuarios

Menús.- Esta opción permite ingresar nuevos mensual sistema, así como modificar sus datos (Figura B.2). Estas opciones funcionan de la misma manera que los ingresos y modificación de registros explicados anteriormente (Apéndice A).

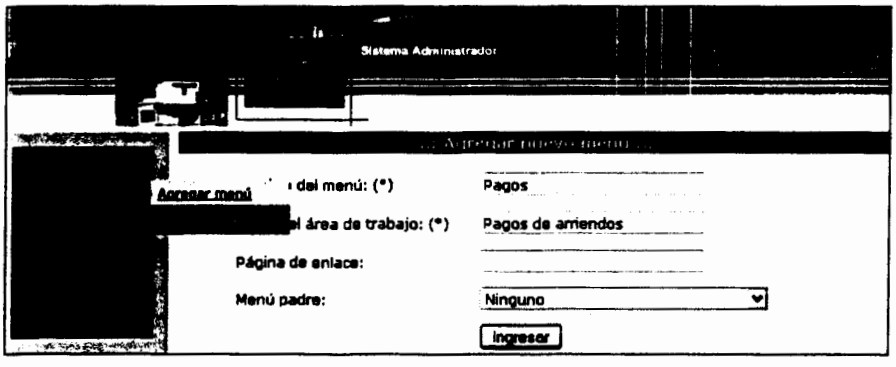


Figura B.2. Pantalla de administración de menús

Formularios.- En esta opción se podrá administrar los contenidos de los formularios que aparecen en las opciones del sistema. El usuario debe seleccionar un menú en el cual desea ingresar un nuevo ítem de su formulario respectivo. Luego deberá escoger el tipo de control Web que va a ingresar,

tras lo cual aparecerán los campos que tienen que ser ingresados, diferentes para cada tipo de control, para añadir el control dentro del formulario del menú seleccionado (Figura B.3). Los tipos de controles que pueden ser ingresados son: cuadros de texto, área de texto, botones, listas, radio botones, casillas de verificación (Checkbox), textos sencillos, y explorador para adjuntar archivos.

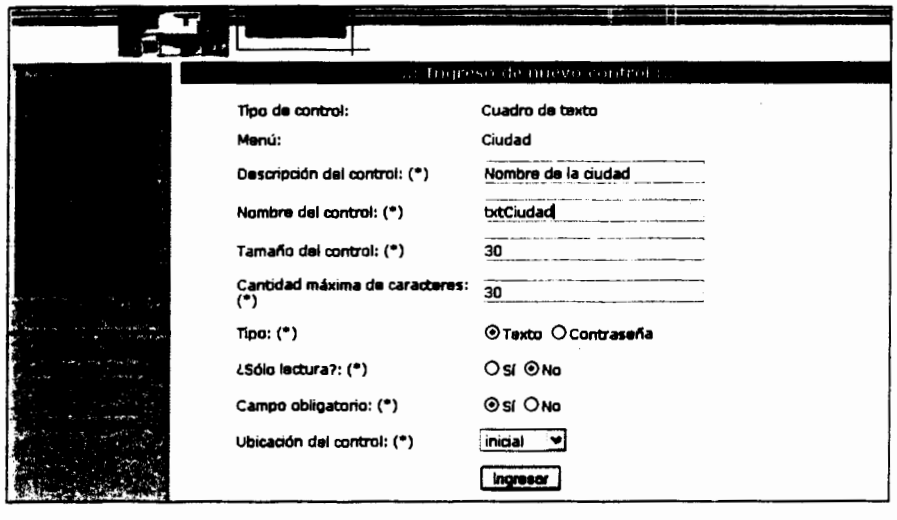


Figura B.3. Pantalla de administración de formularios

Permisos.- Esta opción permite la administración de las opciones que serán presentadas en el área de menús a los usuarios del sistema dependiendo del grupo al que pertenezcan. En primer lugar se debe seleccionar el grupo de usuarios y el menú en el que se van a establecer los permisos (Figura B.4). Luego aparecen los submenús asociados al menú seleccionado; las opciones que tienen el visto son las que el grupo mencionado tiene acceso, pudiendo el administrador seleccionar nuevas opciones o quitar el acceso a otras (Figura B.5).

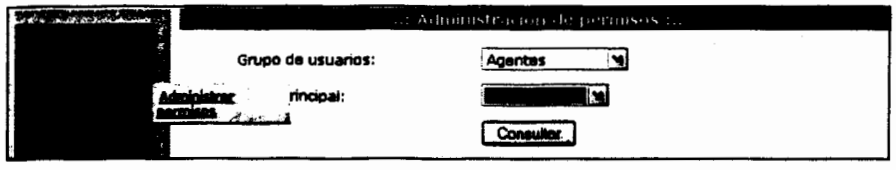


Figura B.4. Pantalla de administración de permisos (Parte I)

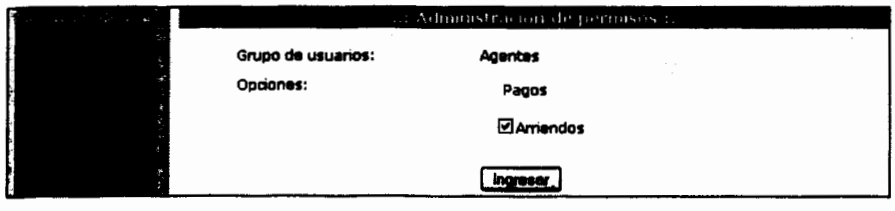


Figura B.5. Pantalla de administración de permisos (Parte II)

Ingresar.- Esta opción permite el ingreso de nuevos registros de: ciudades, sectores de ciudades, tipos de inmueble y tipos de listados. Esta sección se explicó en el Apéndice A.

Buscador.- Esta opción permite el ingreso, modificación o eliminación de los elementos que aparecen en el buscador (Figura B.6). Estas opciones funcionan de la misma manera que los ingresos, modificaciones y eliminaciones de registros explicados anteriormente.

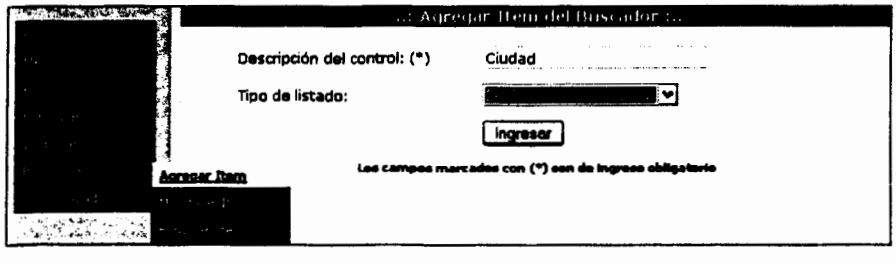


Figura B.6. Pantalla de administración de elementos del buscador

Sincronizador.- Esta opción permite la configuración de una base de datos desde la cual se replican los datos (base de origen), y de bases de datos de destino (hacia donde van los datos sincronizados). Los usuarios pueden añadir dichas bases, modificar los datos de configuración y eliminar alguna de las bases de destino luego de seleccionarlas (Figura B.7).

En estas opciones son necesarios ingresar el host (nombre del servidor o dirección IP), el puerto (opcional, por defecto es 5432), nombre de la base de datos en PostgreSQL, nombre del usuario y contraseña para acceder a la base de datos.

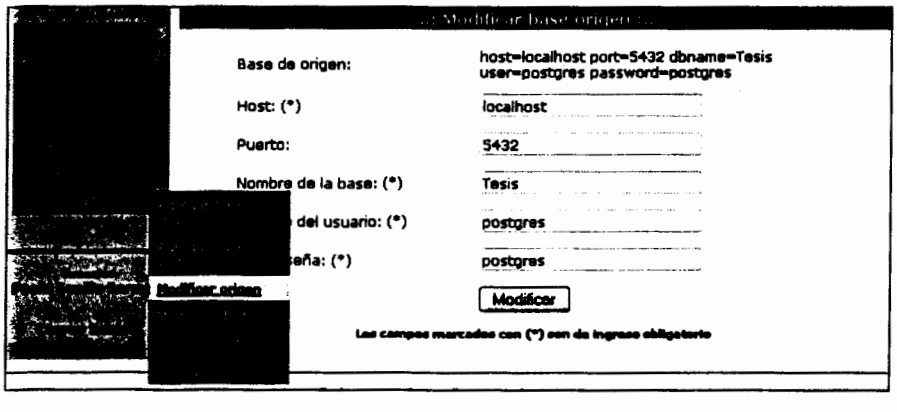


Figura B.7. Pantalla de configuración del servicio de sincronización de datos

Para la configuración de las bases de datos de origen y destino, se ha creado el archivo sincronizador.config, el cual almacenará los parámetros ha seteado como bases de origen y destino. Cabe recalcar que la primera configuración pertenece a la base de origen, y las siguientes a los diversos puntos de destino. Adicionalmente, el usuario sólo podrá configurar la base

de origen una vez, pudiendo modificarla cuando lo requiera. Con respecto a las bases de destino, el usuario puede modificarlas o eliminarlas, según lo considere, para su esquema de trabajo.

A continuación se detalla un esquema de almacenamiento de los parámetros de las bases.

```
host=localhost port=5432 dbname=Tesis user=postgres password=postgres
host=192.168.0.2 port=5432 dbname=Tesis user=postgres password=postgres
```

host = Contiene el nombre o la dirección IP del servidor hacia donde se va a establecer la conexión con la base de datos. Si la base de datos se encuentra hospedada en la máquina local, el valor del argumento será "localhost".
port = Establece el número del puerto hacia el cual está definida la conexión a la base de datos.
dbname = Indica el nombre de la base de datos dada por el recurso conexión de PostgreSQL.
user = Especifica el nombre de usuario para acceder a la base de datos especificada en dbname.
password = Especifica la contraseña para acceder a la base de datos especificada en dbname.

Adicionalmente la opción Sincronizador, permite configurar las tablas que pueden sincronizar sus datos, añadiendo o modificando las mismas en un formato establecido.

Requerimientos del sistema

Hardware

- Procesador: Intel PIII 700 MHz.
- Memoria RAM: 128 Mb.
- Espacio en disco duro: 500 Mb.

Software del servidor (bajo el cual se trabajó esta tesis)

- Servidor Web Apache
- Base de Datos PostgreSQL
- Módulo de PHP (en caso de que se requiera modificaciones en el código fuente)

Instalación del sistema

Para que el sitio Web del sistema entre en funcionamiento se deben copiar todos los archivos que conforman el sitio en el directorio destino dentro de cada uno de los servidores que conforman el negocio. Cabe recalcar que los terminales sólo requieren poseer el acceso al sitio Web a través de un browser, sin necesidad de tener instalado algún componente adicional. Todas las transacciones serán ejecutadas en los servidores. De igual forma, los usuarios públicos sólo requieren de un navegador Web para acceder a los contenidos del sitio.

APÉNDICE C

ADMINISTRADORES DE CONTENIDOS

Los sistemas administradores de contenidos (Content Management System, CMS) son aplicaciones hospedadas en un servidor Web que permiten la creación, administración, distribución y publicación de información, lo que posibilita administrar el contenido de un sitio con facilidad y conveniencia. Los CMS emplean una interfaz que no requiere de conocimientos específicos de desarrollo Web para su manejo y en general se basan en una plantilla diseñada de antemano que actúa como una plataforma para cada página del sitio Web [4].

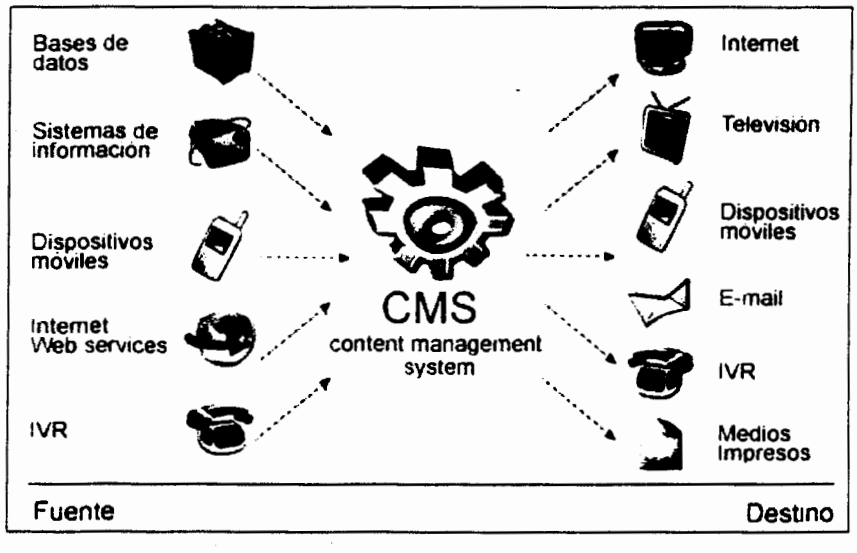


Figura C.1. Flujo de información a través de un CMS.

La utilización de esta herramienta, ahorra personal especializado y tiempo en el ingreso de nuevo material en un sitio Web y permite al usuario concentrarse en el contenido y no en la parte técnica de la elaboración de un sitio, la cual es realizada por programadores.

Además, una amplia gama de contenidos pueden ser publicados a través de un CMS, como páginas simples o complejas con información dinámica que proviene de bases de datos, manuales en línea, documentos generales de negocios, etc.

Las compañías que requieren el uso de un CMS por lo general son organizaciones en donde la publicación en Web está distribuida en muchos sitios y en la cual comunicar el contenido entre los mismos consume mucho tiempo, o que poseen un sitio Web en donde se requiere de frecuentes actualizaciones de contenido y estructura. También es empleado por organizaciones que poseen una fuerte integración de contenido entre el sitio Web y los distribuidores, call centers, boletines de noticias por e-mail o en otros canales de comunicación [4].

Niveles de usuarios en un CMS

Un sistema administrador de contenidos permite determinar cuatro niveles de usuarios [4]:

- **Usuarios públicos,** son aquellos usuarios que tienen las facilidades para poder acceder y navegar a través del sitio Web, y para quienes el único requerimiento es que el contenido sea organizado de una manera apropiada y amigable, y de preferencia, personalizada a sus exigencias, para lo cual es indispensable proveer de motores de búsqueda, consistencia, y una buena estructura de navegación del sitio.

- **Usuarios miembros,** son aquellos que pueden escoger su flujo de trabajo dentro de la sección de contenido. Los miembros pueden sugerir eventos a través de un formulario localizado en una sección pública del sitio Web.

- **Autores y editores,** son aquellos que elaboran el contenido del sitio utilizando un CMS como una herramienta poderosa, eficiente y fácil de usar, que les proveen la capacidad de administrar miles de páginas, publicadas en diferentes plataformas y formatos, a través de plantillas previamente construidas para incorporar contenido. Los autores del contenido no requieren entender HTML o cualquier otra información técnica.

- **Administradores,** son aquellos que administran todo el contenido del sitio Web, con privilegios que incluyen roles administrativos, ciclos de vida, flujos de trabajo, categorías y tipos de contenidos, y administra-

ción del sistema de publicación. También son responsables de la creación de sitios, adición de nuevos usuarios, definición de tipos de usuarios y configuraciones de seguridad del sitio, así como son los encargados del mantenimiento de la estructura e integridad del sitio.

Administración y publicación de la información

En términos generales, la estructura de un administrador de contenidos permite: almacenar y administrar de manera integral los ciclos de la información, incluyendo versiones de documentación, correcciones, aprobaciones; integrar diversos niveles de usuario, de acuerdo a sus roles y responsabilidades corporativas o institucionales; e incorporar reportes en la administración de la información.

De la misma manera, un administrador de contenidos permite publicar y distribuir el contenido hacia una variedad de medios de información como la Internet, haciendo parte de estrategias en el desarrollo de portales; televisión, mediante sistemas de votación en tiempo real; dispositivos móviles, con aplicaciones finales tales como consultas de sistemas empresariales, campañas de marketing y sistema de noticias para operadores móviles; integración de contenidos en tiempo real para su distribución mediante correo electrónico; entre otros.

Ventajas del uso de un administrador de contenidos

En cualquier sistema administrador de contenidos se tienen diferentes herramientas que permiten obtener sistemas que trabajan eficientemente, llevando con ello ventajas tales como [4]:

- Optimizar diversos procesos como la frecuencia de actualización de los contenidos de un sitio o la definición de privilegios en la administración del sistema para editar y publicar la información.

- Organizar y gestionar de manera efectiva el contenido que se publicará en el sitio Web, a través de un sistema basado en plataforma Web y que obtiene la información de una base de datos.

- Manejar la parte técnica de la programación que posee un sitio Web a través del administrador de contenidos, lo que permite que cualquier persona no técnica y con autorización pueda actualizar el sitio, lo cual reduce los tiempos y costos de mantenimiento ya que la mayor parte de estas operaciones pueden ser automatizadas.

- Proporcionar a un sitio consistencia en el diseño de todas las páginas y crea elementos de navegación para los visitantes del sitio.

Los administradores de contenidos poseen características tales como [4]:

- Administración de usuarios, grupos de usuarios y derechos de lo que se les permite a los diferentes tipos de usuarios.
- Fácil incorporación de bases de datos y tablas existentes.
- Facilidad para agregar o editar la información.
- Facilidad en subir y manejar imágenes.
- Configurable a las características particulares de un negocio.

APÉNDICE D

REPLICACIÓN Y SINCRONIZACIÓN DE DATOS

La replicación de datos consiste en la transmisión de los mismos entre dos o más instancias de servidores, permitiendo transferir la información a más de un sitio. Su principal ventaja radica en aumentar la disponibilidad de los datos y mejorar el rendimiento de las consultas globales. La sincronización de datos es un mecanismo similar que, a diferencia de la replicación que opera en el log de cada base de datos, afecta directamente a la capa de datos, con lo que se reduce la complejidad en el desarrollo de la aplicación, pero crea una responsabilidad adicional en el manejo de la seguridad de los datos [1].

Componentes de la replicación

Son representados utilizando metáforas obtenidas de la industria de la publicación [7]. El modelo se compone de los siguientes objetos:

- **El publicador.-** Es un servidor de datos que los coloca a disposición de otros servidores para poder replicarlos.
- **El distribuidor.-** Es un servidor de datos que almacena la base de datos de distribución, los datos históricos y transacciones

- **El suscriptor.-** Es un servidor de datos que actúa como receptor de los datos replicados.
- **La publicación.-** Es el conjunto de artículos que posee una base de datos, especifica un conjunto de datos relacionados lógicamente y los objetos de la base que se desea replicar.
- **El artículo.-**Puede ser una tabla de datos, un procedimiento almacenado, una vista o una función definida por el usuario.
- **La suscripción.-** Es una petición de copia de datos o de objetos de la base de datos que se requieren replicar. Determina cuál es la publicación que será recibida, dónde y cuándo. Pueden ser de inserción o de extracción. El publicador (en las suscripciones de inserción) o el suscriptor (en las suscripciones de extracción) solicitan la sincronización o distribución de datos de una suscripción.
- **Agente de replicación.-** Es el proceso encargado de transmitir y copiar los datos entre el publicador y el suscriptor.

El publicador detecta cuáles son los datos que han cambiado cuando ocurre la replicación transaccional y contiene la información acerca de todas las publicaciones del sitio.

Resolución de conflictos

Los conflictos ocurren continuamente en sistemas Cliente – Servidor, pero

debido a la existencia de mecanismos de bloqueo que actúan como niveles de aislamiento proporcionados por los gestores de bases de datos, facilitan en gran medida su solución inmediata.

Los conflictos pueden producirse al subir registros desde las bases de datos remotas a la consolidada. Cuando se han realizado modificaciones en registros compartidos por más de un usuario y estas variaciones llegan al sincronizador, éste debe decidir cuál debe ser el estado correcto final del registro.

El sincronizador, para detectar los conflictos, debe recibir en la subida de cada registro, el estado actual del mismo en la base remota y el estado en el que se encontraba en dicha base luego de la última sincronización. El sincronizador detecta una colisión cuando encuentra diferencias entre el estado actual de la base consolidada con el estado que recibe de la remota en su última sincronización.

Algunas posibilidades del lado del servidor, para resolver el conflicto anteriormente planteado son:

- Informar y dejar a criterio del usuario la resolución del mismo, presentándole algunas opciones.
- Ignorar el conflicto e informar en un registro de conflictos lo que ha

ocurrido. Un proceso batch podría realizar esto.

- Cancelar la actualización.
- Dar prioridad a la primera modificación.
- Dar prioridad a la última modificación.
- Dejar en manos de un algoritmo genérico la resolución del conflicto sin requerir la acción del usuario.
- Introducir reglas funcionales según el rol/usuario, tabla, fila y columna en la que se produzca el conflicto.

El desarrollo de las reglas en el lado del cliente tendrá que adecuarse a las reglas del servidor. Para que el sincronizador pueda detectar conflictos, el cliente debe subir el registro de la última sincronización así como el actual. Para que el cliente logre conocer qué registros se han modificado y por tanto deben subirse, debería comparar las columnas de los registros de la última sincronización con las actuales y conocer así cuáles han sufrido cambios.

Tipos de replicación

a) Replicación sincrónica.- La transmisión de datos, es decir la copia de éstos, se desarrolla en el momento de la sincronización y consistencia. Esto significa que si se altera la información de una de las bases de datos, esta alteración debe ser replicada a los otros servidores en el momento de la transacción. Los servidores que replican información cooperan entre sí utili-

zando estrategias sincronizadas y protocolos especializados de replicación, que se encuentran involucrados en un proceso llamado ejecución de dos fases (*two phase lock and conmit protocols*), para mantener coherencia entre los conjuntos de datos replicados. Se lo conoce como de dos fases ya que cuando una transacción es replicada hacia un sitio secundario, se necesita de la confirmación de la transacción en el servidor de contingencia para dar por realizada la transacción y poder proceder con la siguiente.

Este esquema es recomendado para aplicaciones en donde la consistencia exacta de los datos es de vital importancia, y su objetivo principal es el de lograr que la pérdida de datos en caso de desastre sea casi nula y poder disponer de una recuperación rápida cuando ocurra la caída de uno de los sitios. Sin embargo, el proceso de ejecución en dos fases puede producir una degradación del rendimiento si las distancias entre los servidores son grandes o si el medio de comunicación utilizado no proporciona un ancho de banda suficiente, por lo que sus costos son mayores.

b) Replicación asincrónica.- Se basa en la captura de todas las transacciones finalizadas en el servidor principal para luego ser aplicadas en los sitios secundarios. Si cambian los datos de una base, dichos cambios serán propagados y aplicados a las otras bases en un segundo paso, dentro de una transacción separada, por lo que la réplica podrá quedar temporalmente fuera de sincronía, pero cuando la sincronización ocurra los datos convergi-

rán para todos los sitios especificados. Existe la posibilidad de configurar la transacción de replicación en intervalos que pueden estar dados en segundos, minutos, horas o hasta en días posteriores.

No necesita de un gran ancho de banda para ejecutarse sobre grandes distancias ni presenta una degradación de importancia en el rendimiento del sistema. Por otra parte, los tiempos de recuperación de la información y la pérdida de datos en caso de un desastre son mayores que los que se pudieran presentar con una replicación sincrónica.

Este tipo de replicación es útil cuando se tienen varios suscriptores que necesitan actualizar sus datos en diferentes momentos. También es de utilidad cuando los suscriptores requieren recibir datos, realizar cambios sin conexión y después sincronizar los cambios con el publicador y otros suscriptores.

BIBLIOGRAFÍA

1. BURETTA, MARIE. Data Replication: Tools and techniques for managing distributed information, John Wiley & Sons, Estados Unidos, 1997.
2. DIX, ALAN; FINLAY, JANET; ABOWD, GREGORY; BEALE, RUSSELL. Human Computer Interaction, Prentice Hall, Gran Bretaña, 1998.
3. ELMASRI, RAMEZ; NAVATHE, SHAMKAT B. Fundamentals of database systems, Addison-Wesley, Estados Unidos, 2000.
4. ERPTODAY.COM. Content Management Tutorial, http://erptoday.com/CMS/Content-Management-Tutorial.aspx.
5. LAU, YUN–TUNG. The Art of Objects: Object–oriented design and architecture, Addison-Wesley, Estados Unidos, 2000.
6. MTBASE – SYBASE DE COLOMBIA. Replicación de Datos y Warm Standby con Sybase Replication Server, http://www.mtbase.com.co/pdf/ha/RS-Stdby.pdf.
7. MONOGRAFÍAS.COM. Replicación de datos en SQL Server, www.monografias.com.
8. URUDATA MONTHLY REPORT. Replicación de datos a nivel corporativo, http://www.cp.com.uy/92/92-replicacion-datos.htm.
9. Técnicas de sincronización de datos, http://www.aui.es/biblio/bolet/bole024/ art_mov_intesys.htm.
10. www.apache.org
11. www.desarrolloweb.com

12. www.php.net

13. www.postgresql.org

14. www.zend.com